UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RODRIGO DE ABREU BATISTA

Busca de Padrões em Séries Temporais

Trabalho de Graduação.

Prof. Dr. Paulo Martins Engel
Orientador

Porto Alegre, dezembro de 2010.

AGRADECIMENTOS

Agradeço a minha família, namorada e amigos pelo apoio incondicional e motivação que sempre recebi e fizeram com que eu não desistisse de chegar até aqui. Agradeço também ao professor Paulo Martins Engel, pela paciência e atenção dedicada ao me orientar nesse trabalho de conclusão, ao professor David da Motta Marques, por ter fornecido os dados que aqui foram utilizados, à professora e colega Cláudia dos Santos Flores, que gentilmente se ofereceu para revisar o presente texto, e aos integrantes da banca, professores João Henrique Flores e Karin Becker, pelas observações e sugestões que contribuíram na melhoria deste trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

SOM	*Self-Organizing Maps*
PIP	*Perceptually Important Points*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O presente trabalho discute e implementa duas abordagens utilizadas na descoberta de padrões em séries temporais: *clustering* e *motifs*. Primeiramente, é apresentada a abordagem de *clustering* com janela deslizante, e os problemas que decorrem da mesma. Na tentativa de contornar os problemas, faz-se uso de Mapas Auto-Organizáveis como alternativa ao *clustering*; por fim, é apresentada a técnica de identificação de pontos perceptualmente importantes, como solução para evitar padrões triviais e reduzir a dimensionalidade da série de entrada. Num segundo momento, é descrita a técnica de descoberta de padrões baseada em motifs, bem como as motivações que levaram a essa abordagem. Uma segunda implementação, usando dessa vez a nova abordagem, é apresentada e, ao final, são comparados os resultados obtidos pela execução dos métodos. Assim, com base nos experimentos realizados, concluiu-se que a abordagem do SOM com janela deslizante gera padrões muito suavizados e que pouco lembram a série original, enquanto que a abordagem por Motifs encontra padrões com melhores resoluções.

Palavras-Chave: Séries Temporais, Descoberta de Padrões, Mapas Auto-Organizáveis, SOM, Motifs, Agrupamento.

ABSTRACT

This work discusses and implements two approaches used in time series pattern discovery: clustering and motifs. Initially the concepts and formalisms related to the time series are introduced, through the techniques used in the search for patterns and concluding with the presentation of implementations, and a comparison of these. At first the approach of clustering with sliding window is presented, and the problems that arise from that. In an attempt to solve these problems, Self-Organizing Maps are used as an alternative to clustering, and finally the technique of identifying perceptually important points is presented, as solution to prevent trivial matches and to reduce the dimensionality of the input series. Secondly, we describe the technique of pattern discovery based on motifs, and the motivations that led to this approach. A second implementation is presented and implemented using this new approach and, finally, the achieved results are compared by the implementation of methods. Thus, based on experiments, it was concluded that the SOM approach with sliding window generates patterns very smooth that does not resemble the original series, while the Motifs approach finds patterns with better resolutions.

Keywords: Time Series, Pattern Discovery, Self-Organizing Maps, SOM, Motifs, Clustering.

1 INTRODUÇÃO

Com a automação de processos e o desenvolvimento de sistemas em áreas como finanças, marketing, ciências econômicas, seguros, demografia, ciências sociais, meteorologia, energia e epidemiologia, entre outros, tem se obtido, além dos produtos decorrentes de cada uma dessas áreas, muita informação associada a elas. Diferentes informações são geradas ao longo das inúmeras etapas associadas aos processos, bem como da observação de fenômenos naturais e experimentos relacionados. Assim, muitos dos dados que hoje são coletados e armazenados, estão associados, de alguma forma, à dimensão tempo.

Surgem então, dessa dimensão adicional, associada aos dados do domínio, inúmeras possibilidades de análise, que vão além das funções básicas como contabilização, cálculo da média e obtenção dos valores máximos e mínimos que ocorrem. Ao se considerar a restrição temporal, os dados passam a formar um conjunto ordenado, chamado série temporal, permitindo que sejam calculadas tendências e observados padrões, possibilitando a descoberta de informações adicionais sobre o domínio do problema.

A Figura 1.1 mostra um gráfico com o valor medido das vazões diárias da Bacia do São Francisco, ao longo de oito anos (1978 a 1986), totalizando 2.920 registros. Através do gráfico, a principal característica que se pode notar é a regularidade e a periodicidade dos registros. A partir dos picos, ou vales, é possível se ter uma ideia de onde começa e termina cada ano. Além disso, por retornar constantemente ao mesmo patamar após as variações apresentadas em Janeiro de cada ano, percebe-se que a série possui tendência decrescente, pois ao observar-se os picos, pode-se notar que os mesmos decrescem.

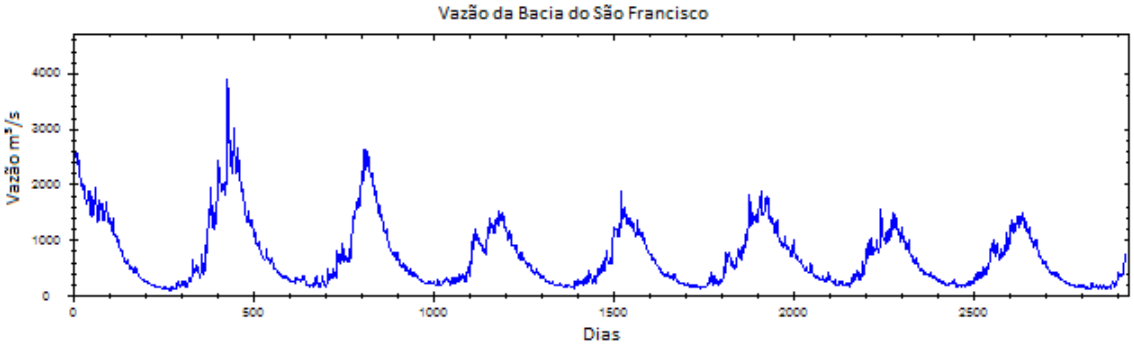

Figura 1.1: Gráfico das vazões do Rio São Francisco entre 1978 e 1986

A correlação presente na amostragem de pontos temporalmente adjacentes restringe, no entanto, a aplicabilidade de muitos dos métodos estatísticos tradicionais, que assumem como premissa que essas observações adjacentes são independentes e identicamente distribuídas (SHUMWAY, 2006). A abordagem utilizada para tratar dos

problemas matemáticos e estatísticos levantados por tais correlações temporais é conhecida por análise de séries temporais.

1.1 Objetivo

O objetivo desse trabalho consiste na realização de um estudo sobre a descoberta de padrões em séries temporais. Tal interesse baseou-se numa demanda iniciada no Instituto de Pesquisas Hidráulicas da UFRGS, onde se deseja responder questões referentes à existência ou não de padrões em séries temporais hidro meteorológicas, bem como a extração daqueles que ocorrem como maior frequência. A descoberta e entendimento de padrões desconhecidos auxiliam no entendimento de fenômenos, na realização de previsões e na economia de recursos decorrente da otimização do próprio monitoramento.

Ambiciona-se, com esse trabalho, primeiramente desenvolver entendimento conceitual e prático sobre as principais abordagens utilizadas na extração de padrões frequentes em séries temporais para que, em um segundo momento, sejam criadas implementações das técnicas que tornem possível a validação e comparação dos resultados.

1.2 Estrutura do Trabalho

O texto encontra-se estruturado da seguinte forma:

- O capítulo 1 apresenta a introdução e a motivação que levou à realização desse estudo;
- O capítulo 2 contém uma introdução às séries temporais, definições e uma abordagem inicial intuitiva sobre como realizar a extração de padrões frequentes em séries temporais.
- O capítulo 3 apresenta uma aplicação de Mapas Auto-Organizáveis como alternativa ao *clustering* para a descoberta de padrões e uma técnica de compressão da entrada baseada em Pontos Perceptualmente Importantes (PIPs).
- O capítulo 4 traz considerações sobre os problemas decorrentes ao se fazer *clustering* com janela deslizante, conforme sugerido no capítulo 3, e introduz uma nova abordagem baseada na busca por Motifs.
- O capítulo 5 apresenta as implementações que foram desenvolvidas, que fazem uso das técnicas abordadas nos capítulos anteriores, bem como um comparativo das características de cada uma.
- Por fim, o capítulo 6 apresenta as considerações finais e os pontos em aberto que podem ser explorados em trabalhos futuros.

2 BUSCA DE PADRÕES EM SÉRIES TEMPORAIS

Antes de iniciar o estudo sobre a busca de padrões em séries temporais, é importante apresentar as definições formais de alguns conceitos e nomenclatura adotadas ao longo deste trabalho. Tentou-se, na medida do possível, manter os conceitos e nomenclatura encontradas na literatura, salvo quando as definições eram conflitantes, onde optou-se pela definição que mais se adaptava à linha seguida nesse estudo.

Definição 1: uma Série Temporal é um conjunto ordenado $T = \{x_1, x_2, \dots, x_n\}$ de n valores reais (Keogh *et al*, 2003).

Duas séries temporais A e B podem, então, ser comparadas calculando-se a distância euclidiana entre elas, conforme (2.1). Tem-se então $Dist(A, B) = 0$, quando A e B forem idênticas.

$$Dist(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \tag{2.1}$$

Assim, dado um conjunto de séries temporais D, todas de mesmo tamanho n, pode-se encontrar as duas séries mais similares como sendo o par de objetos do conjunto que possui menor distância entre si. Mais formalmente, $\forall_{\text{a,b,i,j}}$ o par $\{T_i, T_j\}$ é o mais similar em D se, e somente se, $dist(T_i, T_j) \leq dist(T_a, T_b),\ i \neq j\ e\ a \neq b$.

2.1 *Clustering* de Séries Completas

Pode-se então pensar em agrupar as sequencias semelhantes em grupos distintos, usando-se a distância euclidiana como critério de similaridade. O algoritmo k-*means* (KEOGH *apud* BRADLEY e FAYYAD, 2009) é um algoritmo de *clustering* que pode auxiliar nesse processo, conforme apresentado na Tabela 2.1. Como parâmetro para o algoritmo, é preciso definir de antemão o valor de k, que é a quantidade de grupos (de agora em diante chamados de *clusters*) a serem gerados. Cada *cluster* corresponde a um vetor com a mesma dimensão das séries temporais em D, e pode ter suas componentes inicializadas com valores aleatórios ou pré-definidos.

Definidos os valores iniciais dos *clusters*, percorre-se o conjunto das entradas D associando a cada série temporal T_i, o *cluster* mais próximo. Os *clusters* atuam então como centro dos grupos, e são recalculados a cada iteração de modo a refletir os objetos associados a eles. O processo se repete, reassociando-se os N objetos ao *cluster* mais próximo, enquanto existirem objetos mudando de grupo.

A técnica de *clustering* em séries temporais distintas, considerando sequências inteiras, é citada na literatura por *whole clustering*, e é idêntico ao *clustering* de objetos discretos.

Tabela 2.1: O algoritmo k-*means*

1. Definir o valor de k
2. Inicializar os centros de k *clusters* (podendo ser aleatório)
3. Decidir a classe de cada um dos N elementos associando-os ao *cluster* mais próximo
4. Recalcular o centro dos clusters considerando os padrões associados
5. Se nenhum dos N elementos mudar de cluster na última iteração, então encerra, caso contrário, volta para 3.

A Figura 2.1 ilustra o processo de formação dos *clusters* no algoritmo k-*means* com $k = 3$, onde os círculos representam o centro dos *clusters* e os quadrados, os objetos que estão sendo agrupados. Na figura, são exibidos os clusters iniciais (1), os grupos encontrados após a associação das entradas aos clusters mais próximos (2), o reajuste dos centros dos *clusters* em relação aos seus respectivos objetos associados (3) e, por fim, os novos *clusters* induzidos após o reposicionamento dos centros (4).

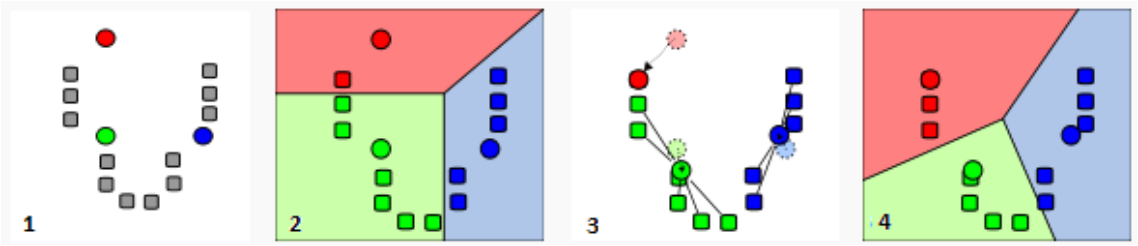

Figura 2.1: Evolução do processo de *clustering* do algoritmo k-*means* (Wikipedia).

2.2 *Clustering* de Subsequências de Séries

Muito frequentemente, deseja-se estudar uma única série temporal com o objetivo de, por exemplo, encontrar os padrões que ocorrem com mais frequência na mesma. Nesse caso, não existe utilidade em comparar a série em questão com outras séries, não cabendo, assim, o *whole clustering* abordado na seção anterior.

O que se deseja vai ao sentido de comparar a série com ela mesma ou com suas partes. Uma abordagem frequentemente usada na literatura consiste em estipular um tamanho w e obter todas as possíveis subsequências da série original com esse tamanho. Tal técnica é chamada de janela deslizante e formalizada nas definições 2 e 3, apresentadas a seguir.

Definição 2: dada uma série temporal T de tamanho n, uma Subsequência S de T é uma amostra de posições contíguas de tamanho $w < n$ em T, isto é, $S = x_p, \ldots, x_{p+w-1}$, para $1 \leq p \leq n - w + 1$ (Keogh *et al*, 2003).

Definição 3: dada uma série temporal T de tamanho n, e uma subsequência de tamanho w, uma matriz D de todas as possíveis subsequências pode ser construída deslizando-se a janela ao longo de T e colocando a subsequência S_p na p-ésima linha de D. O Tamanho da matriz D é $(n + w - 1) \times w$ (Keogh *et al*, 2009).

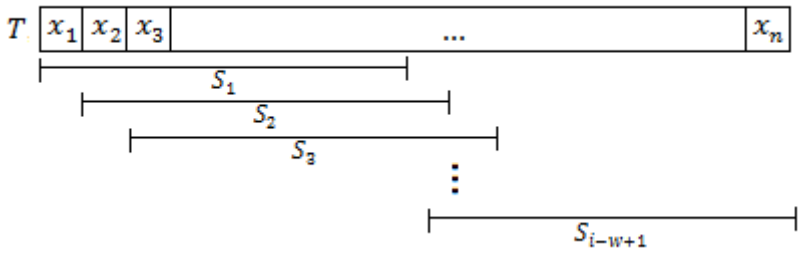

Figura 2.2: Série temporal original T e a decomposição em suas subsequências S_i.

Após a obtenção do conjunto de entrada, representado pela matriz D (Figura 2.3), contendo todas as subsequências de tamanho w da série original, é intuitivo usar o mesmo princípio que motivou o *whole clustering* na seção anterior, a fim de agrupar as subsequências semelhantes. Um padrão representativo e aproximado das subsequências associadas a cada *cluster* pode ser obtido, então, nas componentes dos vetores que representam os centros dos *clusters*.

x_1	x_2	x_3	x_4	x_5	x_6
x_2	x_3	x_4	x_5	x_6	x_7
x_3	x_4	x_5	x_6	x_7	x_8
x_4	x_5	x_6	x_7	x_8	x_9
x_5	x_6	x_7	x_8	x_9	x_{10}
x_6	x_7	x_8	x_9	x_{10}	x_{11}
x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}
x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}
x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}

Figura 2.3: Matriz formada por subsequências de tamanho 6 obtidas pelo método da janela deslizante.

3 UTILIZANDO REDES SOM NA BUSCA DE PADRÕES

Entre os algoritmos de *clustering* existentes – tais como hierárquico, *nearest-neighbor* e k-*means*, talvez o mais popular seja esse último. No algoritmo k-*means*, o número de *clusters k* é um parâmetro crítico a ser definido, que refletirá a quantidade de padrões encontrados. Em se tratando da descoberta de padrões desconhecidos em séries temporais, não é tarefa fácil afirmar *a priori* sobre a existência de um padrão, sendo ainda mais difícil especificar a quantidade de padrões distintos *k* que espera-se encontrar.

Os Mapas Auto-Organizáveis (do inglês, *Self-Organizing Maps* – SOM), oferecem uma solução parcial para o problema de fixar a quantidade de padrões, com a introdução de redundância. Um mapa auto-organizável é um algoritmo de *clustering* baseado em rede neural de aprendizado não supervisionado (CHUNG, 2001) e introduzido nas seções seguintes.

3.1 MAPAS AUTO-ORGANIZÁVEIS

Um Mapa Auto-Organizável é uma rede neural de aprendizado não supervisionado usada para produzir representações de baixa dimensionalidade de um espaço de entrada multidimensional. Nesse processo de reduzir a dimensão dos vetores, tal técnica cria uma rede que mantém a topologia do conjunto de treinamento numa estrutura mais simplificada – tipicamente uni ou bidimensional – chamada de mapa.

3.1.1 Estrutura

Como em uma rede neural, o SOM é construído tendo o neurônio artificial como a unidade mais básica de processamento. Cada um dos nós (neurônios) da rede possui uma posição topológica específica (uma coordenada x e y, por exemplo) e um vetor de pesos com a mesma dimensão que os vetores de entrada (Figura 3.1).

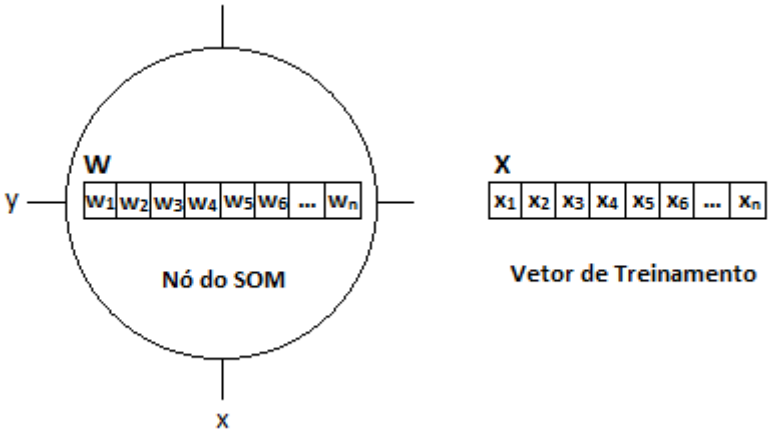

Figura 3.1: Estrutura do nó básico do SOM.

Os nós do SOM são organizados em estruturas que podem corresponder a mapas unidimensionais, bidimensionais, n-dimensionais, variando de acordo com a aplicação. Na figura 3.2 pode-se observar um SOM com 16 nós, organizados em um mapa bidimensional (4x4). As linhas claras conectando os neurônios servem apenas para representar a vizinhança de cada um, não correspondendo a qualquer ligação estrutural entre os mesmos. Adicionalmente, cada um dos nós da rede é ligado aos vetores de entrada (vetores de treinamento).

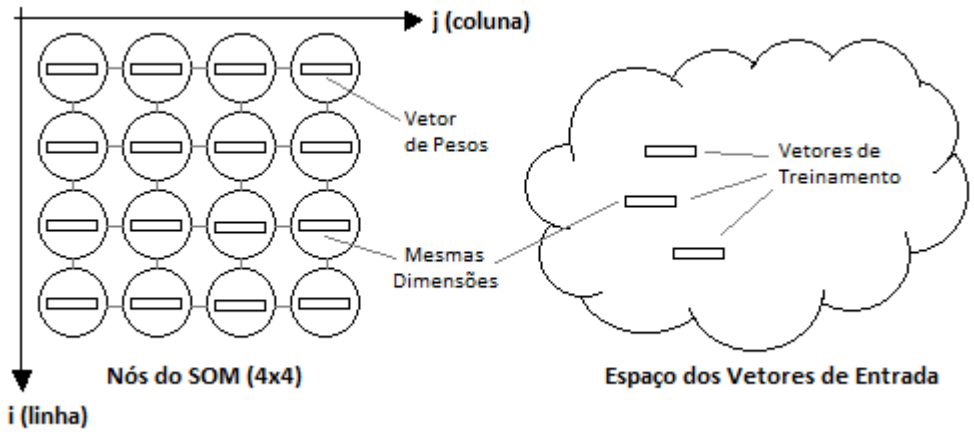

Figura 3.2: Exemplo de uma rede SOM bidimensional 4x4.

3.1.2 O Algoritmo de Aprendizagem

O aprendizado no SOM é baseado no aprendizado competitivo onde os nós de saída da rede competem entre si para serem ativados. A formação do mapa ocorre em duas etapas: a estruturação inicial, que organiza os padrões de entrada no mapa, e a convergência final, responsável pelo refinamento. A segunda etapa leva mais tempo que a primeira, e é caracterizada pelos valores pequenos da taxa de aprendizado. Como muitas iterações através do conjunto de treinamento podem ser necessárias, costuma-se também estipular um máximo de iterações como condição de término do algoritmo.

Durante a fase de aprendizado, calcula-se a distância de cada um dos nós para o padrão apresentado e identifica-se o vencedor. Isso permite especificar qual região do mapa será responsável por aprender o novo padrão.

O algoritmo de treinamento do SOM é exibido na tabela 3.1, e ocorre basicamente em quatro etapas:

Tabela 3.1: Algoritmo de treinamento básico do SOM

Definir parâmetros da vizinhança

Definir taxa de aprendizado

Inicializar os pesos

Repetir

  Para cada padrão de entrada

    Para cada nó j, calcular a distância:

$$D(j) = \sqrt{\sum_i (w_{ij} - x_i)^2}$$

    Encontrar o índice j tal que $D(j)$ é mínimo (vencedor)

    Para cada nó j, e para todo peso i:

$$Fator = e^{-\left(\frac{Dist\_Vencedor}{R_v}\right)}$$

$$w_{ij}(t+1) = w_{ij}(t) + eta \times Factor \times [x_i - w_{ij}(t)]$$

    Reduzir taxa de aprendizado (eta)

    Reduzir raio da vizinhança (R_v)

    Se $eta < eta\_min$

$$eta = eta\_min$$

    Se $R_v < R\_min$

$$R_v = R\_min$$

Até (Convergência ou Número máximo de iterações atingido)

3.1.2.1 Inicialização dos Pesos

Antes do treinamento da rede, os pesos (w_i) dos nós do SOM devem ser inicializados. A inicialização dos pesos pode se dar através da atribuição de valores aleatórios ou, caso alguma informação referente à distribuição dos agrupamentos estiver disponível, receberem valores que reflitam um conhecimento prévio sobre a estrutura do mapa. Nas implementações utilizadas ao longo deste trabalho, cada peso é inicializado com um valor aleatório normalizado ($0 < w < 1$).

3.1.2.2 Escolha do Vencedor

O nó vencedor é o nó da rede que possui pesos que mais se aproximam do vetor de entrada. Tal nó pode ser encontrado percorrendo-se todos os nós da rede e, para cada um, calculando-se a distância euclidiana entre os pesos do mesmo em relação ao vetor

de entrada. O nó com a menor distância $Dist(j)$ é o vencedor, calculado pelas equações 3.1 e 3.2, e será o foco da área do mapa que receberá os ajustes.

$$Dist(j) = \sqrt{\sum_{i}(w_{ij} - x_i)^2} \tag{3.1}$$

$$vencedor = \min_{j \in S} Dist(j) \tag{3.2}$$

3.1.2.3 Determinação da Vizinhança Local do Vencedor

O nó vencedor é calculado a cada iteração, e em seguida calcula-se quais nós pertencem à sua vizinhança local. Isso pode ser feito de forma simples tomando-se os vizinhos imediatamente adjacentes ou de forma mais elaborada levando-se em consideração um raio R, que decresce ao longo das iterações, respeitando a topologia da rede.

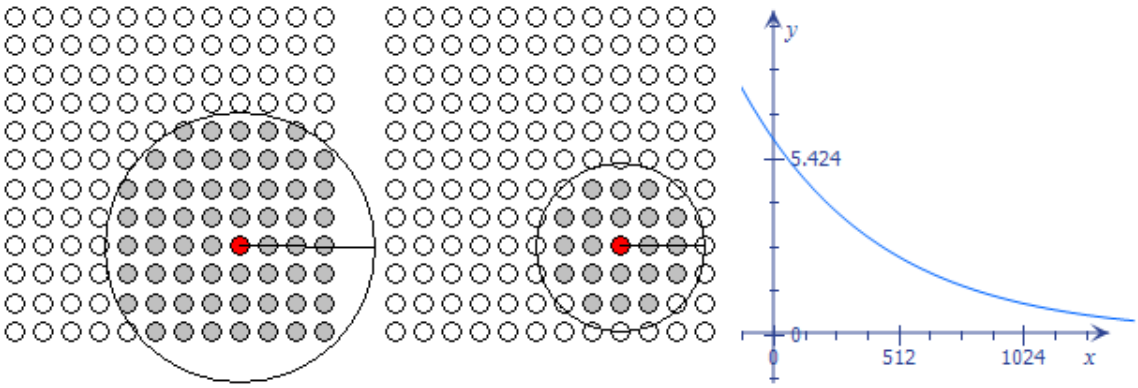

Figura 3.3: Raio da vizinhança decrescendo ao longo das iterações (esquerda e centro) e gráfico da função de decaimento do raio da vizinhança para uma grade de $R_v = 6$ e um total de 1000 iterações.

Para fazer o raio da vizinhança decrescer ao longo do tempo, pode-se usar uma função exponencial decrescente (3.3). Tal função resulta em valores para R_v (raio da vizinhança) que variam entre R_{mapa}, resultante da primeira iteração, e 1, resultante do valor limite de iterações (imagem 3.3 direita).

$$R_v(t) = R_{mapa} \times e^{-(\frac{t}{\lambda})} \quad t = 1,2,3 \ldots \tag{3.3}$$

$$\lambda = \frac{nIterações}{\ln(R_{mapa})} \tag{3.4}$$

3.1.2.4 Ajuste dos Pesos

Todos os nós dentro da vizinhança do vencedor têm seus pesos ajustados de acordo com a equação 3.5, onde t representa o passo de tempo (índice da iteração) e eta representa a taxa de aprendizado calculada pela equação (3.6). No cálculo de eta, λ refere-se à constante de tempo calculada em (3.4), e faz com que o valor resultante da expressão decresça, enquanto que $Fator$ é um coeficiente calculado de modo a fazer com que o efeito do aprendizado (ajuste) seja proporcional à distância que um nó está do vencedor. Assim, o nó vencedor recebe a maior parte do aprendizado ($Fator = 1$), enquanto que nos limites da vizinhança, o efeito do ajuste dos pesos é quase nulo, similar à função exponencial mostrada na imagem 3.4.

$$w_{ij}(t+1) = w_{ij}(t) + eta \times Fator \times [x_i - w_{ij}(t)] \tag{3.5}$$

$$eta = eta_{inicial} \times e^{-\frac{t}{\lambda}} \quad t = 1,2,3\ldots \tag{3.6}$$

$$Fator = e^{-(\frac{Dist\_Vencedor}{R_v})} \tag{3.7}$$

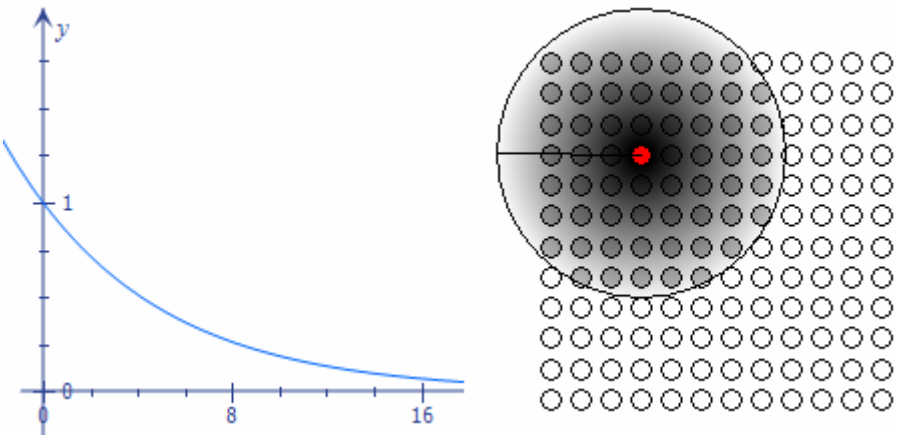

Figura 3.4: Decaimento do fator multiplicativo dentro do raio da vizinhança.

3.2 Aplicando o SOM à Descoberta de Padrões

Para a extração dos padrões que ocorrem mais frequentemente em uma série temporal T, aplica-se o SOM sobre o conjunto de subsequências D, geradas a partir da janela deslizante de tamanho w sobre T. O SOM atua então como um algoritmo de *clustering*, agrupando padrões temporais similares dispersos na série (CHUNG, 2001).

A rede treinada, após o processo de aprendizado (*clustering*) agrupa um conjunto de padrões $p_1, \ldots, p_k$, onde K é o número de nós na camada de saída, podendo ocorrer redundância. Durante o processo de treinamento, os padrões temporais, representados

por cada nó da camada de saída, são ajustados a cada iteração, de acordo com o padrão apresentado na entrada. Ao final, uma estrutura que respeita a topologia das entradas apresentadas e capaz de representar grande parte dos nós vencedores é formada. Esse conjunto dos padrões, obtidos na camada de saída, é uma aproximação dos padrões que aparecem com maior frequência na série original (CHUNG, 2001). É importante notar que os padrões foram encontrados levando-se em conta uma janela deslizante de tamanho fixo *w* e, portanto, tais padrões possuem tal tamanho.

A figura 3.5 exemplifica a formação dos padrões em dois nós distintos em um processo de treinamento do SOM.

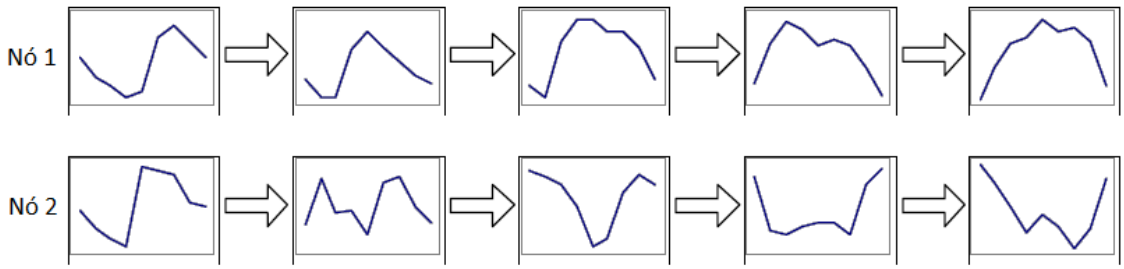

Figura 3.5: Exemplo da descoberta de dois padrões durante a fase de aprendizado (CHUNG, 2001).

Como a camada de saída do SOM possui tamanho fixo, é comum se obter mais padrões do que aparentemente possui a série original. Nesse caso, padrões similares ou redundantes são percebidos, principalmente para pequenos valores de *w*. Assim, torna-se interessante um processo de remoção de redundância, capaz de agrupar padrões similares e representá-los como um padrão mais geral.

3.3 Processo de Remoção de Redundância

O processo de remoção de redundância é um refinamento para o algoritmo de aprendizagem do SOM, apresentado na tabela 3.1; e tem como finalidade reduzir a quantidade de padrões encontrados, mantendo apenas os mais significativos. Isso é feito em duas etapas:

A primeira delas consiste em percorrer novamente o conjunto das entradas e identificar, para cada uma, qual dos padrões encontrados que a representa melhor (maior similaridade); um contador das ocorrências é mantido para cada padrão e, ao final do processo, removem-se os padrões que não tiveram ocorrências.

A segunda etapa, responsável por agrupar padrões similares e redundantes (tabela 3.2), consiste em calcular a distância entre todos os padrões encontrados e agrupar os mais semelhantes em um novo padrão. Isso é feito respeitando-se uma constante de diferença máxima tolerada λ, que é a distância máxima permitida entre dois padrões para que o agrupamento entre eles ocorra. O novo padrão é então calculado respeitando-se o peso que cada um dos padrões tinha sobre o conjunto de subsequências (quantidade de ocorrências). O processo é repetido enquanto existirem pares de padrões com distância menores que λ.

Tabela 3.2: Remoção de Redundância em Padrões Encontrados (CHUNG, 2001).

Início

Definir parâmetro λ, que é a máxima diferença tolerada para que um agrupamento ocorra

Repetir

Calcular as distâncias entre todos os padrões

$$Dist(i,j) = \sqrt{\frac{\sum_{k=1}^{w}(a_{i,k}-a_{j,k})^2}{w}}, para\ i = 1\ a\ K\ e\ j = 1\ a\ K, onde\ i \neq j$$

Encontrar a menor distância $\min(Dist(i,j))$

Se $\min(Dist(i,j) \leq \lambda$, então agrupar os padrões i e j como um novo padrão, representado por:

Para $k = 1$ até w, calcular:

$$a_{novo,k} = \left(a_{i,k} \times \frac{ocorr_i}{ocorr_j + ocorr_j}\right) + \left(a_{j,k} \times \frac{ocorr_j}{ocorr_i + ocorr_j}\right)$$

Número de padrões $K = K - 1$

Até $(\min(Dist(i,j)) > \lambda\ ou\ K = 1)$

Fim

3.4 Representação da Série de entrada

Quando a série temporal a ser analisada possuir um grande volume de dados, pode ser conveniente utilizar uma representação alternativa que reduza a dimensão da mesma. O método mais simples para isso consiste em amostrar a série de entrada a uma taxa de n/m (KEOGH apud ASTROM, 1969), onde n é o comprimento da série original, e m é o comprimento da série que se busca após a redução (Figura 3.6 superior). O ponto-fraco desse método é a característica de distorcer muito a série de dados quando usada uma taxa de amostragem muito baixa.

Como aperfeiçoamento desse método, sugeriu-se usar o valor médio de cada segmento para representar o correspondente conjunto de pontos (FALOUSTOUS, 2000; KEOGH, 2000). Isso produz uma série de dados que aproxima melhor o conjunto original, com a mesma quantidade de pontos do método anterior (Figura 3.6 inferior).

Outros métodos foram propostos, como a utilização de segmentos de tamanhos não fixos, adaptados ao formato da série (KEOGH, 2001), ou a aproximação da série por pontos perceptualmente importantes (CHUNG, 2001a). Por não ser o foco deste estudo, o aprofundamento em técnicas de representação de séries temporais, o modelo escolhido, para ser utilizado foi o dos Pontos Perceptualmente Importantes (do inglês, *Perceptually Important Points* – PIPs). A técnica é descrita na próxima seção.

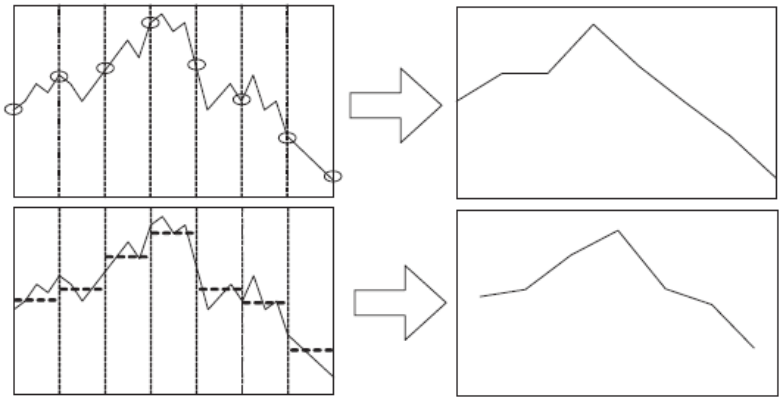

Figura 3.6: Aproximação da série temporal original por amostragem (figura superior), e aproximação baseada na média dos segmentos (inferior).

3.4.1 Aproximação baseada em Perceptually Important Points (PIPs)

Na busca por padrões em séries de dados, são pontos específicos que caracterizam determinados comportamentos, diferenciando-os do que seria aparente ruído aleatório. Esses padrões são identificados por apenas alguns pontos críticos, caracterizados pelos vales e picos (mínimos e máximos locais). Assim, para fins de análise e mineração de dados, tais pontos tornam-se especialmente importantes, pois é através deles que o processo de identificação humano busca padrões ao analisar gráficos. Em aplicações financeiras, onde a análise de padrões em séries de dados desenvolveu-se significativamente, muitos dos padrões consagrados são referenciados por nomes, como é o exemplo do *Head and shoulders* (Figura 3.7 esquerda), e são classificados basicamente pela identificação de tais pontos (Figura 3.7 direita).

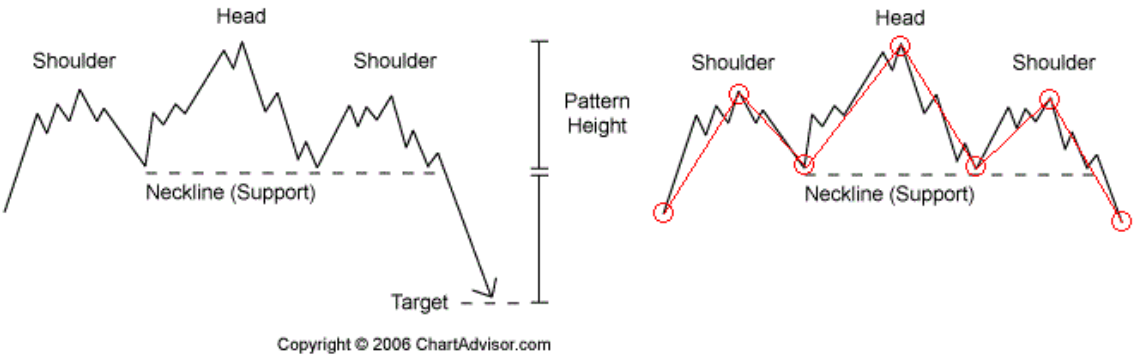

Figura 3.7: Padrão head and shoulders (cabeça e ombros) esquerda (LANGAGER, 2006) e a correspondente aproximação por PIPs (direita).

A técnica de aproximar uma série baseando-se nos pontos mais notáveis foi primeiramente introduzida por Chung *et al* (FU, 2010) e usada para *matching* de padrões em aplicações financeiras. Os autores batizaram esses pontos críticos de *Perceptually Important Points* (PIPs) e a técnica consiste na ideia de localizar Q pontos críticos em uma série P, conforme a tabela 3.3.

Tabela 3.3: Algoritmo para a identificação de PIPs.

```
Função LocalizaPIP (P, Q)
      Entrada: sequência P[1..n], tamanho de Q
      Saída: padrão sp[1..m]
Início
      sp[1] = p[1], sp[m] = p[m]
      Repetir
            Selecionar o ponto p[j] com a máxima distância dos
            pontos adjacentes em sp (sp[1] e sp[m] inicialmente)
            Adicionar p[j] a sp
      Até (sp[1..m] esteja preenchido)
Fim
```

Ao início do processo, os dois primeiros PIPs são o primeiro e o último pontos da sequência original P. O próximo PIP é o ponto em P que possui maior distância em relação aos dois PIPs anteriormente encontrados. O quarto PIP será o ponto em P com maior distância em relação a dois PIPs adjacentes, podendo ocorrer tanto no intervalo entre o primeiro e o segundo PIPs, quanto no intervalo entre o segundo e o último. O processo continua até que a quantidade especificada de Q pontos seja localizada.

Para se determinar a distância de um ponto p_3, pertencente a P, em relação a seus PIPs adjacentes, Chung *et al* (2001) sugere calcular a distância vertical entre p_3 e a reta que liga os PIPs (Figura 3.8). A equação para o cálculo da distância vertical VD entre um ponto $p_c(x_c, y_c)$ e a reta que liga os pontos $p_1(x_1, y_1)$ e $p_2(x_2, y_2)$ é apresentada em (3.8).

$$VD(p_3, p_c) = |y_c - y_3| = \left| \left(y_1 + (y_2 - y_1).\frac{x_c - x_1}{x_2 - x_1} \right) - y_3 \right| \tag{3.8}$$

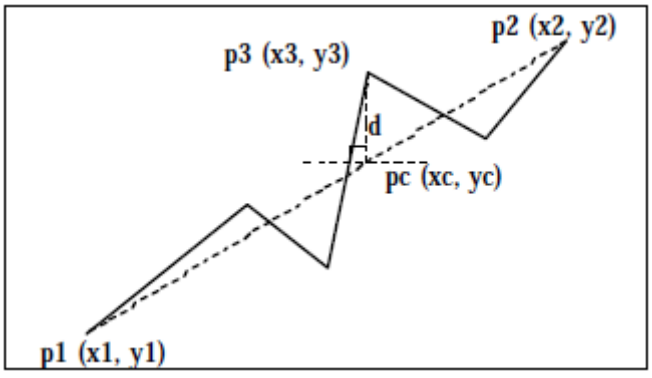

Figura 3.8: Cálculo da Distância Vertical (VD) (CHUNG, 2001).

A figura 3.9 ilustra passo-a-passo o processo de identificação dos PIPs que formam um padrão *Head and soulder*. No exemplo, a sequência P, contendo 29 pontos, foi aproximada por 7 PIPs.

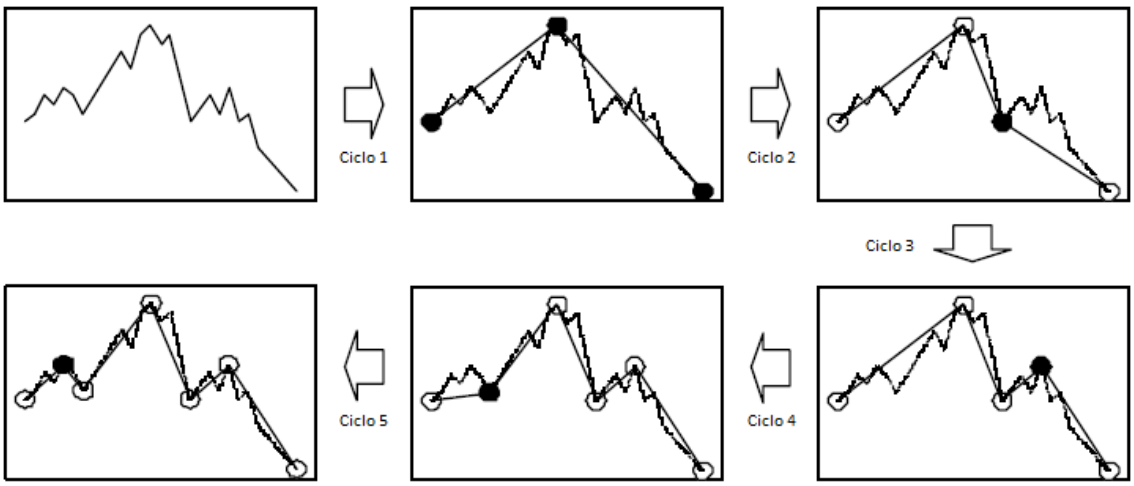

Figura 3.9: Identificação de 7 PIPs (padrão head and shoulder).

Após a identificação dos PIPs na sequência de dados, a similaridade (distância) entre a sequência gerada SP e uma dada sequência a ser comparada Q, pode ser calculada utilizando (3.9).

$$Dist(SP, Q) = \frac{1}{n}\sum_{k=1}^{n}(sp_k - q_k)^2 + \frac{1}{n-1}\sum_{k=1}^{n-1}(sp_k^t - q_k^t)^2 \qquad (3.9)$$

Na equação anterior sp_k^t e q_k^t representam, respectivamente, as coordenadas temporais dos pontos sp_k e q_k da sequencia. A primeira parcela da soma em (3.9) refere-se ao cálculo da média das diferenças dos valores originais da série (valores do domínio), enquanto que a segunda parcela é responsável por considerar as diferenças existentes nas coordenadas temporais. A introdução da segunda parcela ao cálculo se faz necessária porque, ao se considerar os PIPs, não mais existe uma correspondência temporal direta entre a subsequência da série original e outra a ser comparada.

3.5 Descoberta Eficiente de Padrões Baseada na Compressão da Entrada

Apesar da possibilidade de se aplicar o processo de *clustering* diretamente sobre os dados obtidos da janela deslizante, isso se torna custoso ao se trabalhar com grande volume de dados. Para se contornar esse problema, é possível utilizar o algoritmo de identificação de PIPs para comprimir os padrões de entrada (CHUNG, 2001). A ideia é substituir os segmentos de dados originais pelos seus PIPs, para que a dimensionalidade da entrada possa ser reduzida.

Outro ponto importante a se observar, ao buscar padrões desconhecidos, é que não se sabe *a priori* o tamanho desses. Fica difícil então definir o tamanho *w* da janela deslizante. Seria razoável, então, que para se descobrir padrões de tamanhos variados, o processo de descoberta fosse conduzido utilizando diferentes valores de *w*. Os resultados seriam então reunidos para se formar um conjunto final dos padrões encontrados.

A fim de se evitar a obtenção de padrões duplicados ou redundantes, oriundos da execução do algoritmo para diferentes tamanhos de janela, assim como a execução de todo o processo de *clustering* para cada resolução *w*, os autores Chung *et al* (2001) sugeriram uma abordagem mais sistemática. Nessa abordagem, batizada de *combined multi-resolution pattern discovery* (Figura 3.10), os autores sugeriram uma etapa de pré-processamento onde são mapeadas entradas de diferentes dimensões *w*, para uma mesma dimensão w_c, através da técnica de compressão baseada em PIPs; com isso, o processo de aprendizado do SOM, é executado somente uma vez e a fase de eliminação de redundâncias, aproveitada.

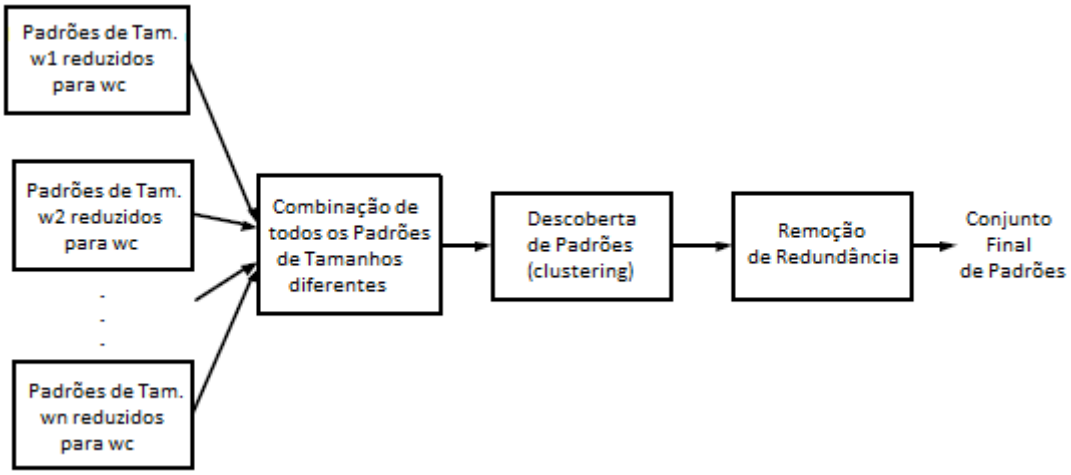

Figura 3.10: Processo de Descoberta de Padrões com Multi-Resolução Combinada (CHUNG, 2001).

4 DESCOBERTA DE PADRÕES BASEADA EM MOTIFS

A abordagem de se utilizar *clustering* na descoberta de padrões em séries temporais sofreu fortes críticas, com a publicação de trabalhos como o de Keogh *et al* (2003) e a posterior comprovação por Idé *et al* (2006). Keogh e Lin (2003) classificaram a técnica de *clustering* baseada em janela deslizante como sendo sem sentido (no original, *meaningless*), chegando a compará-la a um algoritmo randômico, onde "a saída independe das entradas" (KEOGH e LIN, 2003). No artigo, os autores vão mais além, classificando da mesma forma o trabalho de muitos autores que utilizaram *clustering* com janela deslizante como base em seus estudos.

Keogh e Lin (2003) afirmam que o problema de tal abordagem não está no algoritmo de *clustering* propriamente dito – por exemplo, k-*means*, *nearest-neighbor*, c-*means*, hierárquico, entre outros. – e sim, na forma como as subsequências de entrada para o algoritmo são obtidas: técnica da janela deslizante. Na a condução do experimento para validar sua teoria, os autores usaram o algoritmo k-*means*, um dos mais popularmente implementados, e a distância euclidiana, a métrica para cálculo de distância mais empregada na área. Uma visão geral do experimento é descrita a seguir.

4.1 O Problema com a Janela Deslizante

Por ser o algoritmo k-*means* uma heurística, nem sempre se obtém o conjunto ótimo de *clusters* em uma dada execução. Uma técnica para minimizar esse problema, é realizar múltiplos inícios e escolher o melhor conjunto de clusters (KEOGH apud BRADLEY e FAYYAD, 1998). Uma questão que surge, então, é saber quanta variabilidade existe entre duas inicializações distintas, que pode ser calculada pela seguinte equação:

- Seja $A = (\bar{a}_1, \bar{a}_2, \dots, \bar{a}_k)$ os centros dos *clusters* obtidos por uma execução do k-*means*.
- Seja $B = (\bar{b}_1, \bar{b}_2, \dots, \bar{b}_k)$ os centros dos *clusters* obtidos por uma segunda execução do k-*means*.
- Seja $dist(\bar{a}_i, \bar{a}_j)$ a distância euclidiana medida entre os centros de dois clusters.

Assim, a distância entre dois conjuntos de *clusters* pode ser definida como:

$$cluster\_dist(A, B) = \sum_{i=1}^{k} \min\left(dist(\bar{a}_i, \bar{b}_j)\right) \quad , 1 \leq j \leq k \tag{4.1}$$

Através da equação, espera-se que cada cluster em A seja mapeado no seu correspondente mais próximo em B. O somatório de todas as distâncias, então, é tomado como medida de similaridade entre os dois conjuntos. Tal medida pode, então, ser usada não somente para comparar os clusters obtidos em diferentes execuções do k-*means* para um mesmo conjunto de dados, como também para avaliar a similaridade dos clusters encontrados em fontes de dados distintas (Keogh e Lin, 2003).

Em seguida, realizaram-se três execuções do k-*means* sobre um conjunto de subsequências obtidas pela técnica de janela deslizante em uma série temporal do mercado de ações. Os conjuntos dos centros dos *clusters* obtidos foram agrupados em um conjunto que se chamou de $\hat{X}$. O mesmo foi feito para uma série temporal gerada por caminho aleatório (*random walk*).

Mediu-se, então, a distância média entre todos os conjuntos de *clusters* pertencentes ao conjunto $\hat{X}$, usando-se a equação (4.2). Também, calculou-se a distância média entre os conjuntos de *clusters* pertencentes a $\hat{X}$ em relação aos conjuntos pertencentes a $\hat{Y}$, usando-se (4.3).

$$dist\_media\_\hat{x} = \frac{\sum_{i=1}^{3}\sum_{j=1}^{3} cluster\_dist(\hat{X}_i, \hat{X}_j)}{9} \tag{4.2}$$

$$dist\_media\_\hat{x}\_\hat{y} = \frac{\sum_{i=1}^{3}\sum_{j=1}^{3} cluster\_dist(\hat{X}_i, \hat{Y}_j)}{9} \tag{4.3}$$

Com isso, os autores chegaram a uma fração que chamaram de "insignificância do *clustering*", apresentada em (4.4). Assim, se para cada execução o processo de *clustering* gera *clusters* similares, não importando as sementes iniciais, o numerador deve ser próximo de zero; enquanto que não existe razão para que sejam similares os *clusters* de dois conjuntos gerados por dados completamente diferentes. Dessa forma, espera-se que o denominador seja um número relativamente grande, fazendo com que o resultado da fração seja próximo de zero, quando $\hat{X}$ e $\hat{Y}$ forem gerados a partir de conjuntos de dados diferentes.

$$insignificância\_do\_clustering(\hat{X}, \hat{Y}) = \frac{dist\_media\_\hat{x}}{dist\_media\_\hat{x}\_\hat{y}} \tag{4.4}$$

Para fins de comparação realizou-se o mesmo experimento, sobre os mesmos dados, porém com subsequências randomicamente extraídas, em vez de obtidas com a janela deslizante. Essa abordagem é equivalente ao *whole clustering* mencionado na seção 2.1. Além disso, a fim de garantir que os resultados obtidos não são consequência de uma combinação específica de parâmetros, usou-se diferentes combinações de k e w, tendo sido cada uma delas repetidas 100 vezes tomando-se a média.

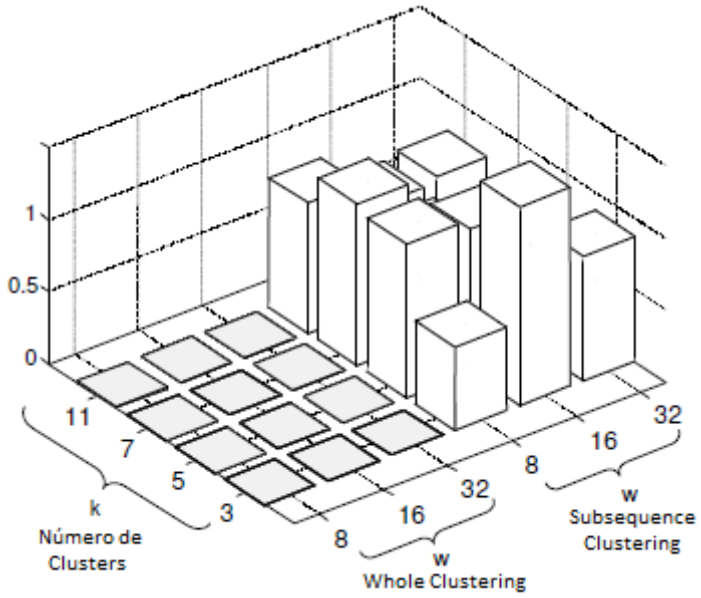

Figura 4.1: Comparação entre *whole clustering* e *subsequence clustering* (KEOGH e LIN, 2003).

Os resultados demonstram que os centros dos *clusters* encontrados pela abordagem de *clustering* com janela deslizante sobre a série do mercado de ações são similares a quaisquer *clusters* encontrados sobre dados randômicos, enquanto que os resultados referentes às execuções com *whole clustering* demonstram uma situação diferente. Posteriormente, os autores repetiram o mesmo experimento para outras séries de dados, assim como para outras abordagens de *clustering*, todas obtendo resultados semelhantes.

Um experimento que auxilia no entendimento do problema pode ser realizado tomando-se, por exemplo, 3 padrões distintos e concatenando-os dezenas de vezes com o objetivo de formar uma longa série temporal. Tal série contém, indiscutivelmente, 3 padrões que se repetem. No entanto, executando-se um k-*means* com $k = 3$, sobre as subsequências obtidas a partir da janela deslizante sobre a série, obtém-se os padrões plotados na figura 4.2.

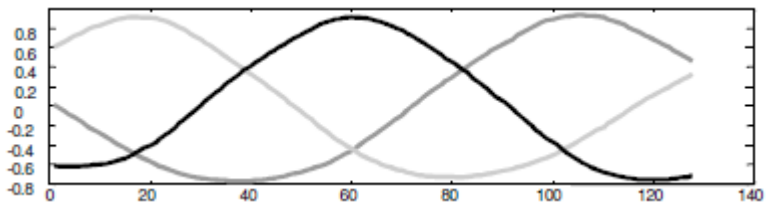

Figura 4.2: Três centros de *clusters* encontrados no *clustering* de subsequências com janela deslizante (Keogh *et al*, 2003).

É importante notar que foram encontrados padrões senoidais, igualmente defasados que, ao se calcular a média de suas componentes, obtém-se um padrão constante. De fato, ao se executar um k-*means* com $k = 1$ sobre qualquer conjunto de subsequências geradas a partir de janela deslizante, o resultado será uma reta constante igual à média da série. Isso ocorre porque, ao se considerar todas as subsequências de uma série

temporal, cada valor x_i da série original ocorre igualmente em todas as posições da janela (Figura 4.3); então, ao calcular-se a média durante o ajuste do centro dos *clusters*, estão se considerando os mesmos valores na obtenção de cada uma de suas componentes.

	A	B	C	D	E	F	G	H
1	1,258	1,252	1,253	1,266	1,267	1,258	1,244	1,238
2	1,252	1,253	1,266	1,267	1,258	1,244	1,238	1,230
3	1,253	1,266	1,267	1,258	1,244	1,238	1,230	1,232
4	1,266	1,267	1,258	1,244	1,238	1,230	1,232	1,236
5	1,267	1,258	1,244	1,238	1,230	1,232	1,236	1,244
6	1,258	1,244	1,238	1,230	1,232	1,236	1,244	1,250
7	1,244	1,238	1,230	1,232	1,236	1,244	1,250	1,251
8	1,238	1,230	1,232	1,236	1,244	1,250	1,251	1,252
9	1,230	1,232	1,236	1,244	1,250	1,251	1,252	1,261
10	1,232	1,236	1,244	1,250	1,251	1,252	1,261	1,284
11	1,236	1,244	1,250	1,251	1,252	1,261	1,284	1,305
12	1,244	1,250	1,251	1,252	1,261	1,284	1,305	1,311
13	1,250	1,251	1,252	1,261	1,284	1,305	1,311	1,306
14	1,251	1,252	1,261	1,284	1,305	1,311	1,306	1,292
15	1,252	1,261	1,284	1,305	1,311	1,306	1,292	1,276
16	1,261	1,284	1,305	1,311	1,306	1,292	1,276	1,263
17	1,284	1,305	1,311	1,306	1,292	1,276	1,263	1,264
$\bar{x}$	1,241	1,241	1,240	1,239	1,239	1,238	1,238	1,237

Figura 4.3: Exemplo de uma matriz composta de subsequências obtidas pela técnica de janela deslizante, onde os valores se repetem em subsequências adjacentes, uma vez em cada posição da janela. A última linha exibe o cálculo da média.

O k-*means* com $k > 1$ gera, então, k ondas senoidais, defasadas de maneira diferente e imprevisível, para cada execução. Uma explicação matemática mais formal e detalhada foge do escopo do presente trabalho e pode ser encontrada em (IDÉ, 2006). Keogh e Jin (2003) concluíram em seu estudo que, ao se usar janela deslizante na obtenção de subsequências, para quaisquer que sejam as séries de entrada, técnica de *clustering* escolhida e valores das sementes para geração dos pesos iniciais, os *clusters* obtidos não serão significativos.

4.2 O Problema dos *Trivial Matches*

Uma subsequência é dita ser um *Trivial Match* quando ela é similar à sua sequência adjacente, formada pela janela deslizante (CHUNG, 2005). A figura 4.4, mostrada a seguir, ilustra tal conceito. De fato, considerando-se uma subsequência S_i pertencente a um *cluster* e, buscando-se por subsequências similares geradas pela janela deslizante, tipicamente espera-se encontrar os melhores matches para S_i como sendo as subsequências $\dots, S_{i-2}, S_{i-1}, S_{i+1}, S_{i+2}, \dots$ (KEOGH, 2003). Ou seja, as subsequências que melhor casam com uma dada subsequência são aquelas onde a janela deslizante foi pouco deslocada. Tal associação caracteriza o que os autores chamaram de *Trivial Match*, como sendo o casamento de uma sequência com suas versões deslocadas, e melhor formalizado pela Definição 4.

Definição 4: dada uma distância R, uma subsequência S, iniciando na posição i, e M e uma sequência a comparar, iniciando na posição j. Diz-se que M é um *Trivial Match* para S de ordem R, se $i = j$ ou não existir uma subsequência M', iniciando em j' tal que $D(S, M') > R$, e tanto $j < j' < i$ ou $i < j' < j$ (Keogh *et al*, 2003).

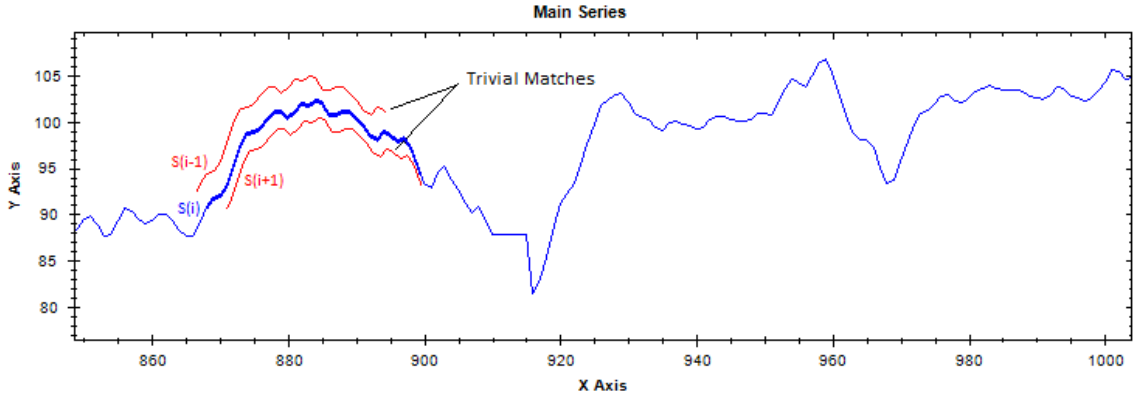

Figura 4.4: Para uma subsequência S_i, os *Trivial Matches* ocorrem nas sequências adjacentes.

Outro fato interessante a se notar é que a quantidade de *Trivial Matches* varia de acordo com as características da série. Por exemplo, uma curva suave, como uma gaussiana, terá mais subsequências adjacentes semelhantes em função da sua lenta velocidade de mudança, do que um padrão com transições bruscas ou ruidosas. A figura 4.5 ilustra esse fenômeno: em A é exibida uma série temporal que foi amostrada com uma janela deslizante de tamanho $w = 64$, enquanto que em B é mostrado um gráfico da quantidade de *Trivial Matches* existentes para cada posição de A. Ainda na figura 4.5, em B é possível notar que a curva mais suave é envolvida por uma quantidade maior de ocorrências triviais do que as curvas quadradas de transições bruscas.

A consideração de tais subsequências – caracterizadas como *Trivial Matches* – durante o processo de *clustering*, é o que Keogh e Lin (2003) apontaram como sendo o motivo pelo qual os centros dos *clusters* degeneram para senóides.

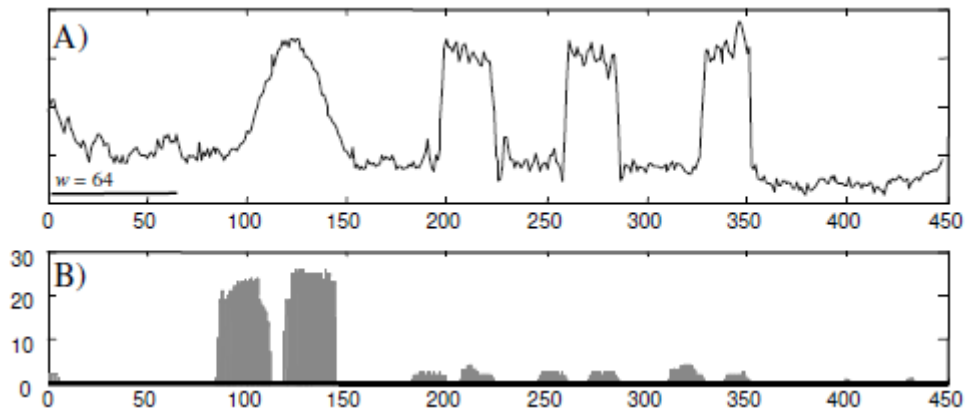

Figura 4.5: A) Uma série temporal T que parece ter 3 ondas quadradas ruidosas. B) Número de *Trivial Matches* para cada S_i em T. $R = 1, w = 64$. (KEOGH, 2003).

4.3 Evitando *Trivial Matches* pela Detecção dos PIPs

Após a publicação do trabalho de Keogh e Lin (2003), e das consequentes críticas ao método de descoberta de padrões em séries temporais baseado em *clustering* e janela deslizante, Chung *et al* (2005) revisitaram seu algoritmo baseado em Perceptually Important Points (CHUNG, 2001a) e propuseram uma melhoria, visando contornar os problemas mencionados. A técnica consiste em evitar considerar os *trivial matches* durante o processo de aprendizado.

Uma vez que já havia sido sugerida a conversão da série original em seus PIPs (CHUNG, 2001), a fim de fazer *clustering* multi-resolução e reduzir o tamanho da entrada, os autores também conseguiram tirar proveito dessa discretização para detectar e desconsiderar matches triviais.

Considerando-se uma série temporal $T = \{x_1, x_2, \dots, x_n\}$ e fixando-se uma largura de janela w, um conjunto de subsequências $D(T) = (S_i = \{x_i, \dots, x_{i+w-1}\} \mid i = 1, \dots, n - w + 1)$ é formada. Para a identificação de *Trivial Matches* durante o processo de obtenção das subsequências, um método baseado na detecção de alterações dos PIPs identificados é introduzido. Comparando-se os conjuntos dos PIPs identificados em duas subsequências distintas, um *Trivial Match* ocorre se o mesmo conjunto de PIPs for identificado para ambas, e a segunda subsequência pode ser descartada (CHUNG et al, 2005). Caso os conjuntos de PIPs sejam diferentes, então ambas sequencias são não-triviais e devem ser consideradas como candidatas à descoberta de padrões.

A figura 4.6 ilustra o processo. Nas subsequências S_1, S_2 e S_3, os mesmos PIPs – sinalizados como elipses na imagem – foram identificados e, portanto, as subsequências S_2 e S_3 devem ser consideradas *Trivial Matches* de S_1. Assim, S_2 e S_3 são filtradas, não seguindo adiante no processo de descoberta de padrões. Já para a subsequência S_4, foi encontrado um conjunto de PIPs diferente dos encontrados para as subsequências anteriores, o que faz com que essa não seja identificada como *Trivial Match.* Então, a subsequência S_4, junto com S_1 é levada adiante no processo de *clustering*. Uma vantagem deste método é que não existem parâmetros a serem definidos que sejam decisivos à classificação de um *Trivial Match*, como é o caso do algoritmo inicial sugerido por Keogh e Lin (2003).

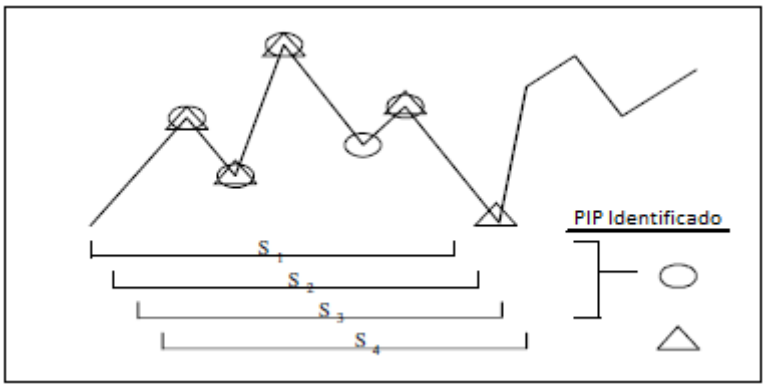

Figura 4.6: Trivial Matches em subsequências pela identificação de PIPs (CHUNG, 2005)

Depois de identificados e removidos os *trivial matches*, as subsequências não-triviais são levadas adiante no processo de descoberta de padrões. Tipicamente serão

usadas como entrada para um algoritmo de *clustering* (i.e. k-*means* ou SOM), como no trabalho de Chung *et al* (2001).

4.4 Contornando o Problema da Janela Deslizante

Uma vez demonstrado que a busca de padrões baseada em janela deslizante gerava *clusters* sem significância, alguns autores tiveram que rever seus métodos e resultados. No mesmo trabalho em que Keogh e Lin (2003) questionaram o método de *clustering* da janela deslizante, os autores sugeriram outra abordagem.

O algoritmo foi motivado por duas observações: a primeira delas é que a tentativa de considerar todas as subsequências no processo de *clustering* leva a padrões que não condizem com a realidade; e a segunda, que considerar os *trivial macthes* causa suavização, levando a padrões pouco detalhados (KEOGH e Lin, 2003). Nesse trabalho, os autores sugerem considerar a busca de *motifs* nas séries temporais. Intuitivamente, motifs são padrões – subsequências – que se repetem em cadeias discretas, como por exemplo, trechos de uma música ou sequências de DNA (KEOGH apud REINERT, 2000). A figura 4.7 ilustra a ocorrência de motifs em uma série temporal, e a Definição 5 formaliza o conceito.

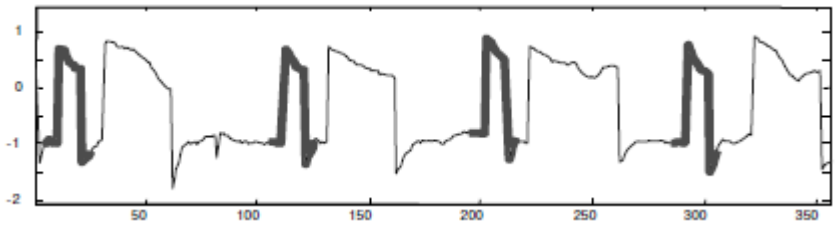

Figura 4.7: Exemplo de ocorrência de motifs em uma série temporal (KEOGH, 2003)

Definição 5: um *motif* de subsequência é o par mais similar de subsequências (S_i, S_j) de uma série temporal T. Formalmente, $\forall_{a,b,i,j}$ o par (S_i, S_j) é um motif para T se, e somente se, $dist(S_i, S_j) \leq dist(S_a, S_b), |i-j| \geq w$ e $|a-b| \geq w$ para $w > 0$ (Keogh *et al*, 2009).

Definição 6: dada uma série temporal T e uma distância R, o grupo de motifs mais significativo em D, chamado 1-Motif, é o conjunto maximal de subsequências onde a distância entre elas é menor que $2R$. Mais formalmente, C_1 é 1-Motif com distância R se, e somente se, $\forall_{S_i,S_j} \in C_1, Dist(S_i, S_j) \leq 2R$ e $\forall_{S_k} \in D - C_1, Dist(S_k, S_j) > 2R$. Assim, o K-ésimo grupo mais significativo de motifs em D, chamado de K-Motif, é o grupo de motifs C_k com a maior quantidade de elementos e que satisfaz $\forall_{S_i,S_j} \in C_k, Dist(S_i, S_j) \leq 2R$ e $\forall_{S_p} \in D - \{C_1, \dots, C_J\}, Dist(S_p, S_j) > 2R$ (Keogh *et al*, 2003)..

A figura 4.8 apresenta um exemplo visual do porque a distância mínima requerida entre dois motifs é $2R$ (Definição 6). Caso a mínima distância requerida fosse apenas R (figura A), então dois grupos de motifs compartilhariam grande parte dos seus elementos. A figura B demonstra que, estando os centros dos motifs distantes a pelo

menos uma distância de $2R$, existe uma garantia que os conjuntos de motifs sejam disjuntos.

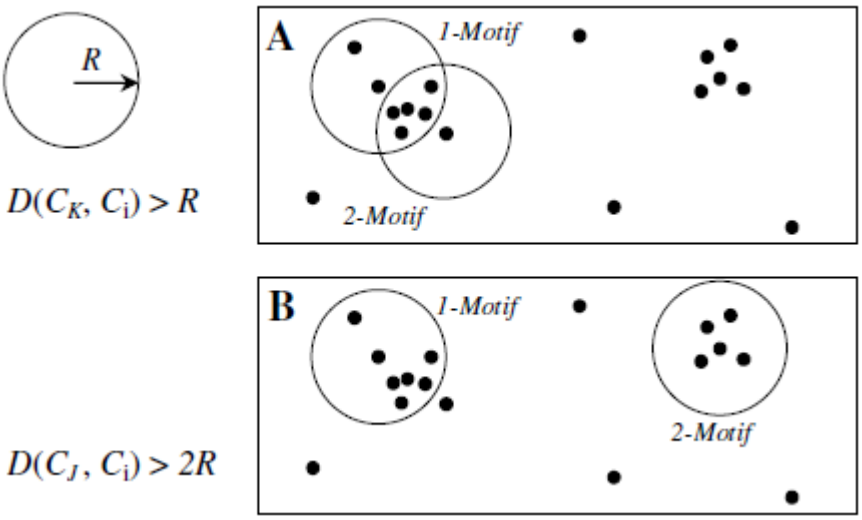

Figura 4.8: Motifs separados por uma distância R em A e $2R$ em B (KEOGH *et al*, 2003)

A definição de um motif já elimina explicitamente os *trivial matches*. Além disso, é possível notar que os motifs definem uma região circular envolvendo um pequeno conjunto dos dados, existindo diversas subsequências que não estão incluídas em nenhum motif (KEOGH, 2003). Apesar de parecer razoável, a pura descoberta dos motifs ainda não é suficiente para encontrar os padrões mais frequentes em séries temporais. Ainda que tenha sido desconsiderado o problema em especificar R, e assumindo que a distância considerada ao encontrar os motifs seja a euclidiana, o algoritmo de detecção dos motifs terá problemas ao encontrar, por exemplo, padrões que foram separados em grupos de motifs distintos, graças ao formato circular das regiões, imposto pela distância euclidiana. Ocorrerão casos de padrões que necessitem de 2 ou mais grupos de motifs para serem corretamente aproximados (KEOGH 2003). A Figura 4.9 ilustra essa ideia.

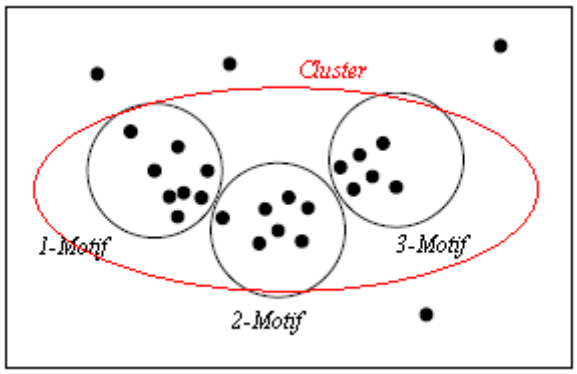

Figura 4.9: Três grupos de motifs detectados que pertencem a um mesmo padrão.

Uma solução sugerida por Keogh e Lin (2003) seria, então, utilizar um algoritmo de *clustering* para agrupar tais subsequências após a detecção dos K-Motifs (Tabela 4.1). Apesar de o autor ter sugerido originalmente o k-*means* ou o *hierarchical clustering* como algoritmo para agrupar os motifs, também podem ser utilizados os mapas auto-organizáveis abordados vistos na seção 3.

Tabela 4.1: Visão geral do algoritmo de cluster baseado em motifs (KEOGH, 2003).

Início Definir o valor de k. Descobrir K-motifs nos dados, para $K = k \times c$ (c é alguma constante na faixa de 2 a 30). Rodar k-means, ou *k partitional hierarchical clustering*, ou qualquer outro algoritmo de *clustering* sobre as subsequências cobertas pelos K-motifs. Fim

4.4.1 Encontrando Motifs

A fim de ilustrar o principio do processo de descoberta de motifs, a Figura 4.10 mostra 8 pontos no plano cartesiano, que podem ser vistos como oito subsequências de tamanho 2 com suas componentes associadas aos eixos. O motif desse conjunto de dados ocorre no par de subsequências (S_4, S_5) e seriam, portanto, subsequências similares.

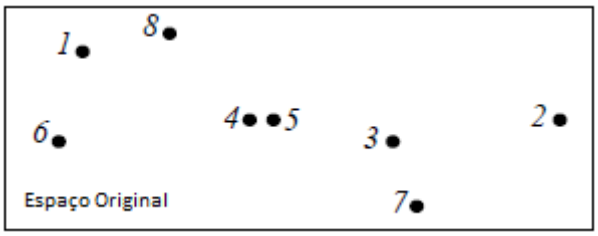

Figura 4.10: Exemplo de motif no plano cartesiano (adaptado de KEOGH et al, 2003).

O problema de se encontrar um motif (S_i, S_j) em um conjunto de subsequências bidimensionais D, plotadas no plano cartesiano, resume-se então a resolver o problema de encontrar o par de pontos mais próximos em um conjunto (*closest-pair problem*). O algoritmo força bruta para encontrar o motif em um conjunto D consta na Tabela 4.2.

Tabela 4.2: Descoberta de Motif por Força Bruta.

```
Início
      Min_dist = infinito
      Para i de 1 até m
            Para j de i+1 até m
                  Se dist(D_i, D_j) < Min_dist então
                        Min_dist = dist(D_i, D_j)
                        S_1 = i
                        S_2 = j
Fim
```

O princípio do algoritmo funciona da seguinte forma: primeiramente assume-se que a menor distância encontrada entre dois pontos (min_dist) é infinita. Em seguida, percorrendo-se todas as $\frac{m(m-1)}{2}$ combinações de pontos, calcula-se a distância entre os pares, mantendo atualizada na variável min_dist a menor distância até então encontrada, bem como os índices S_1 e S_2, que representam o par onde ela ocorre.

Saindo do plano cartesiano e estendendo-se o mesmo princípio para um espaço n-dimensional, as subsequências, antes representadas como pontos com componentes x e y, passam a ter n componentes. Assim, generalizando-se a função de cálculo da distância euclidiana para n componentes (ver equação 2.1), obtém-se facilmente o algoritmo.

4.4.2 Otimizando a busca por Motifs

Como o algoritmo compara a combinação de todos os pares de objetos na busca pela menor distância, seu custo é quadrático $O(m^2)$ no número de elementos. Adicionalmente, estendida a função do cálculo da distância entre objetos para n componentes, esta passou a ter custo linear $O(n)$ no tamanho das subsequências, o que deixa o algoritmo força bruta com uma complexidade $O(m^2 n)$.

Em Keogh *et al* (2009) é apresentado um algoritmo otimizado para a busca de motifs (tabela 4.3). O algoritmo, apesar de ainda possuir complexidade $O(m^2)$ para o pior caso, ganha desempenho ao realizar podas significativas no espaço de busca, evitando o cálculo da distância para todos os pares.

Tabela 4.3: Algoritmo MK Motif Discovery (KEOGH *et al*, 2009).

```
Função MK_Motif (D, R)
     Entrada:  conjunto de subsequências D,
               número de pontos de referência R
     Saída: índices L1 e L2 das subsequências em D
Início
     Best_so_far = INF
     Para i = 1 até R
          ref_i = subsequência randomicamente escolhida D_r em D
          Para j = 1 até m
               Dist_{i,j} = dist(ref_i, D_j)
               Se Dist_{i,j} < best_so_far
                    Best_so_far = Dist_{i,j}
                    L1 = r
                    L2 = j
          S_i = desvio_padrao(Dist_i)

     Encontrar uma ordenação Z para os índices das
     subsequências em ref tal que S_{Z(i)} ≥ S_{Z(i+1)}

     Encontrar uma ordenação I para os índices das
subsequências em D tal que Dist_{Z(1),I(j)} ≤ Dist_{Z(1),I(j+1)}

     Offset=0, abandon = false
     Enquanto abandon = false
          Offset = offset + 1, abandon = true
          Para j = 1 até m
               Reject = false
               Para I = 1 até R
                    Lower_bound = |Dist_{Z(i),I(j)} − Dist_{Z(i),I(j+offset)}|
               Se Lower_bound > best_so_far
                    Reject = true, break
               Senão Se i = 1
                    Abandon = false
          Se reject = false
               Se dist(D_{I(j)}, D_{I(j+offset)}) < best_so_far
```

```
                        Best_so_far = dist(D_I(j), D_I(j+offset))
                        L_1 = I(j)
                        L_2 = I(j+offset)
Fim
```

A estratégia dos autores consiste em tomar R subsequências randômicas como referência e calcular a distância dessas para todos os demais pontos. A subsequência mais bem distribuída – com menor desvio padrão – é então tomada como referência no estabelecimento de um limite inferior para a distância, parâmetro decisivo no descarte dos candidatos a pares de motifs.

A poda no espaço de busca é feita, então, por desigualdade triangular (4.5), de modo que, tendo-se os dois valores à esquerda da inequação, é possível calcular-se um limite inferior para $dist(D_i, D_j)$ com apenas uma operação de subtração (KEOGH *et al*, 2009).

$$dist(ref, D_i) - dist(ref, D_j) \leq dist(D_i, D_j) \tag{4.5}$$

Caso ocorra de o limite inferior ser menor que $best\_so\_far$ – o mínimo corrente – é possível descartar o par (D_i, D_j) com segurança, caso contrário, (D_i, D_j) continua sendo um candidato a motif.

5 IMPLEMENTAÇÃO DAS TÉCNICAS ABORDADAS

Com o objetivo de validar e experimentar os conceitos e técnicas vistas, duas implementações foram desenvolvidas ao longo do desse trabalho: uma primeira, utilizando-se das técnicas de janela deslizante para a geração das subsequências, e do SOM como algoritmo de *clustering* (CHUNG, 2001); e outra, utilizando a descoberta de padrões baseada em Motifs, sugerida em Keogh e Lin (2003).

A linguagem de programação escolhida para o desenvolvimento foi o C#, por possuir um ambiente de desenvolvimento conveniente para a criação de interfaces e geração de gráficos para análise.

Os dados utilizados para validar os experimentos foram cedidos pelo Instituto de Pesquisas Hidráulicas da UFRGS, e consistem em informações de nível da lagoa do Taim coletados a cada hora, durante o ano de 2009 (figura 5.1).

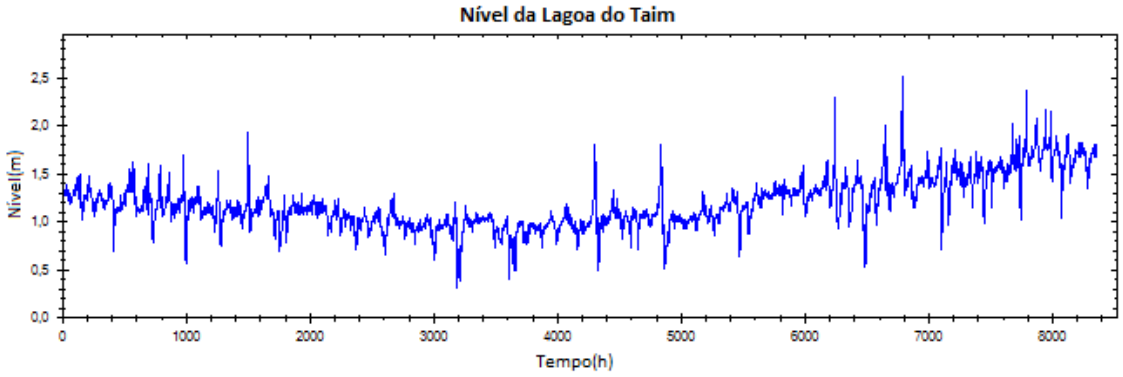

Figura 5.1: Nível da lagoa do Taim em 2009.

5.1 Uma Implementação usando Mapas Auto-Organizáveis (SOM)

A primeira implementação desenvolvida como experimento, baseou-se no trabalho de Chung *et al* (2001) e utilizou a técnica de janela deslizante em conjunto com o algoritmo de mapas auto-organizáveis. Nessa abordagem, a janela deslizante é utilizada sobre a série temporal de entrada para a obtenção das subsequências fornecidas como entrada para o algoritmo. Nesse caso, o SOM atua como um algoritmo de *clustering*, agrupando e ajustando subsequências similares, a fim de se obter os padrões.

5.1.1 Interface

Optou-se por desenvolver a interface das implementações utilizando componentes que automatizassem a geração de gráficos. Isso facilitou a análise e interação, minimizando o uso de outros softwares. Na interface da aplicação (Figura 5.2), a série carregada para análise é plotada na região superior (1), enquanto que os gráficos inferiores exibem o comportamento dos parâmetros raio da vizinhança (2) e taxa de aprendizado (3), respectivamente.

Na coluna à direita na interface (4) são exibidos os padrões encontrados pelo SOM. Ao se colocar o mouse sobre um padrão, subsequências da série principal correspondentes àquele padrão são destacadas. Como mencionado no capítulo 2, muitos padrões similares e redundantes podem ser encontrados. Então, o parâmetro Lambda, situado no canto superior direito da janela (5), pode ser ajustado pelo usuário a fim de agrupar padrões similares.

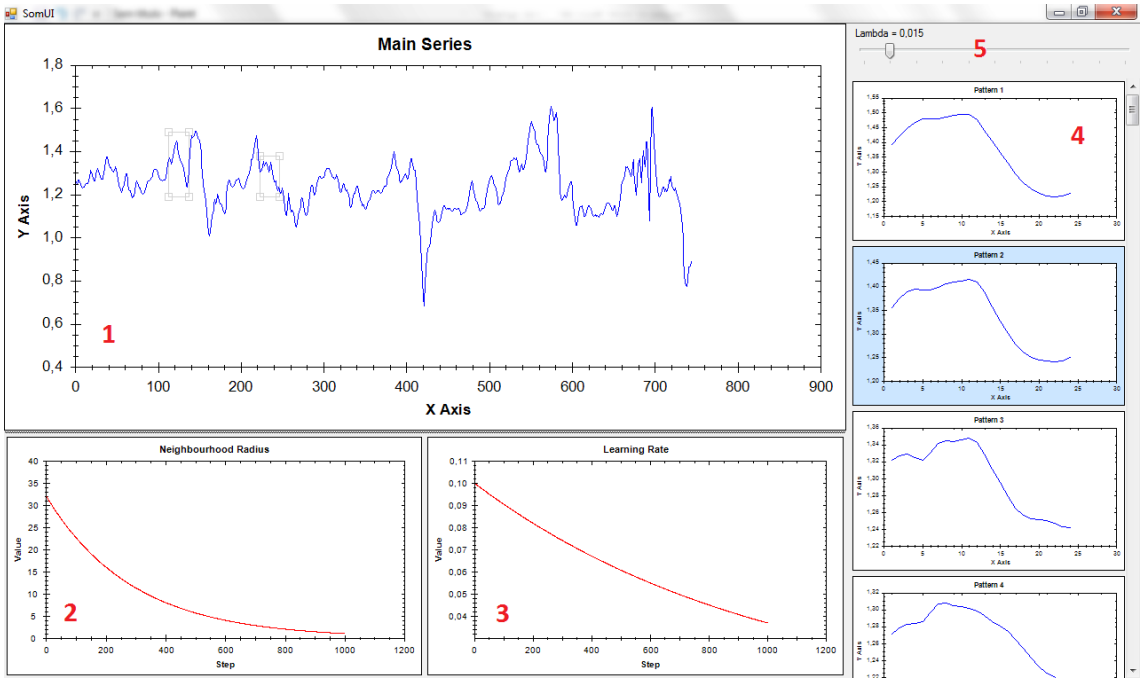

Figura 5.2: Apresentação da interface da primeira implementação.

Adicionalmente, a figura 5.3 mostra a disposição dos padrões encontrados na estrutura de saída do SOM. Os padrões em questão foram gerados processando-se o algoritmo para uma série de $n = 760$ pontos (737 subsequências) e um SOM com grade bidimensional 8x8, janela $w = 24$ e 1000 iterações sobre o conjunto de treinamento. É interessante observar no mapa gerado, que cada padrão é similar aos seus vizinhos, diferindo, em muitos casos, em um pequeno deslocamento da janela deslizante. Isso sugere o problema apontado por Keogh e Lin (2003). Tais padrões, por serem muito semelhantes, são agrupados na etapa de remoção de redundância dando origem a curvas cada vez mais suavizadas, com baixo detalhamento e pouca utilidade para fins de análise.

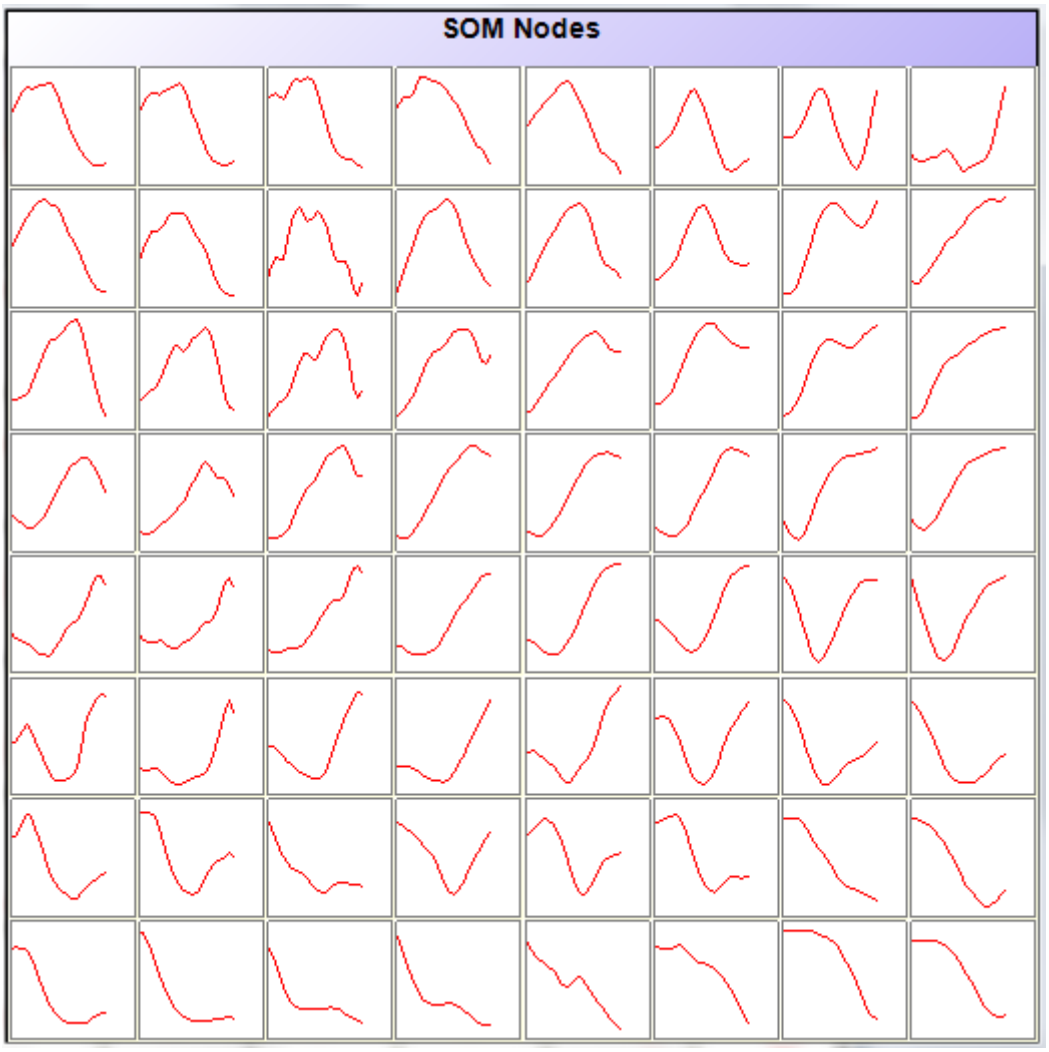

Figura 5.3: Padrões resultantes nos nós do SOM 8x8 após 1000 iterações.

Com o objetivo de contornar o problema dos padrões pouco significativos encontrados, modificou-se a implementação, incorporando a abordagem de detecção de *trivial matches* sugerida em Chung *et al* (2005). Isso foi realizado modificando-se o processamento da janela deslizante de forma a realizar a identificação dos PIPs para cada subsequência, comparando com as anteriores, conforme explicado na seção 4.3. Modificou-se, também, a interface anterior a fim de permitir a visualização e a manipulação dos parâmetros, conforme exibido na Figura 5.4. Nessa, além da série principal (1) e dos padrões encontrados (4), podem ser definidos os parâmetros de discretização da série em PIPs (2), os parâmetros de treinamento do SOM (3), coeficiente de agrupamento de padrões (5) e o botão de início de execução (6). O tempo decorrido no treinamento do SOM é mostrado no canto inferior esquerdo (7).

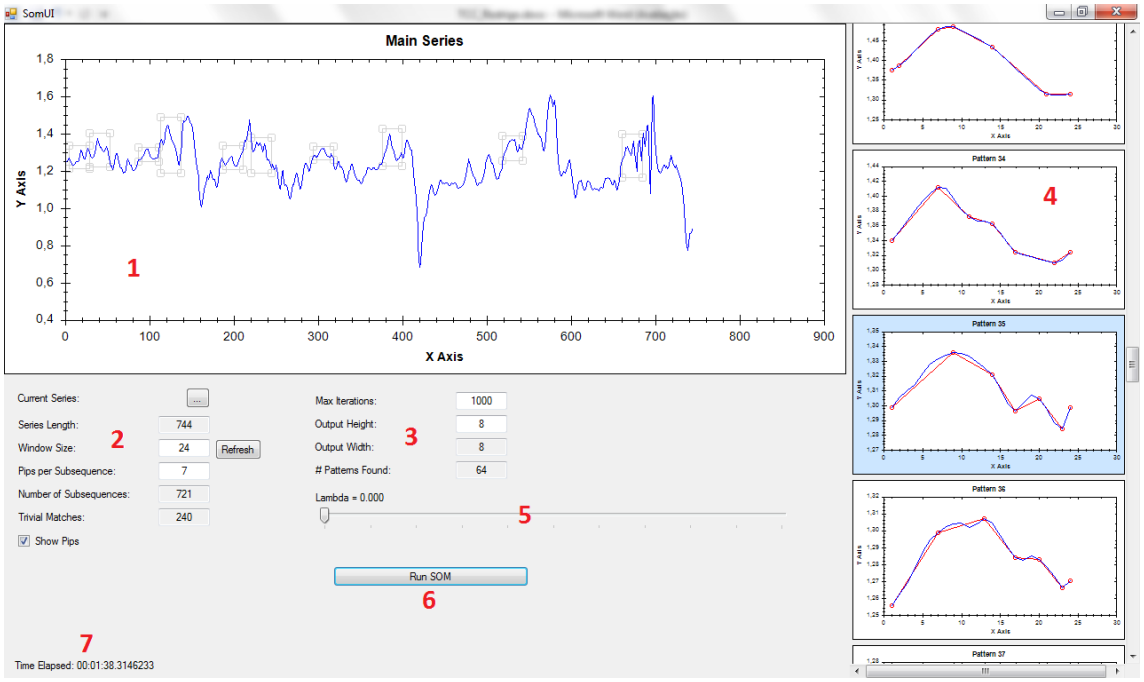

Figura 5.4: Interface modificada para a detecção de *Trivial Matches* pela identificação de PIPs.

5.2 Uma Implementação baseada em Motifs

Na segunda implementação realizada, tomou-se por base o trabalho de Keogh e Lin (2005), onde os autores introduziram o conceito de motifs aplicado à busca de séries temporais, e o de Keogh et al (2009), onde buscou-se um algoritmo eficaz para encontrar tais motifs. Ainda que a abordagem utilize janela deslizante para a geração das subsequências, evita os *trivial matches* ao introduzir uma distância mínima que separa uma subsequência de outra, conforme descrito na seção 4.4. Então, para cada tamanho de janela, é possível encontrar o par de subsequências mais semelhante (Motif).

5.2.1 Interface

Para a exploração dos Motifs, realizou-se também uma implementação que possibilitasse a visualização das séries, com controles que permitissem o ajuste dos parâmetros do algoritmo, facilitando o estudo exploratório. A figura 5.5 exibe o resultado. Na parte superior é mostrada a série principal (1), com molduras destacando a ocorrência dos motifs. Na parte inferior esquerda, são sobrepostos os motifs encontrados, a fim de facilitar a comparação (2). Em (3) é mostrado um gráfico contendo duas séries: a primeira, em vermelho, exibe a distância do Motif encontrado para cada tamanho de janela; enquanto que a segunda, em azul, exibe o desvio padrão das diferenças das subsequências que compõem o motif. A parte inferior da janela contém controles para definir o tamanho da janela *w* (4) e a distância mínima *R* entre K-motifs (5).

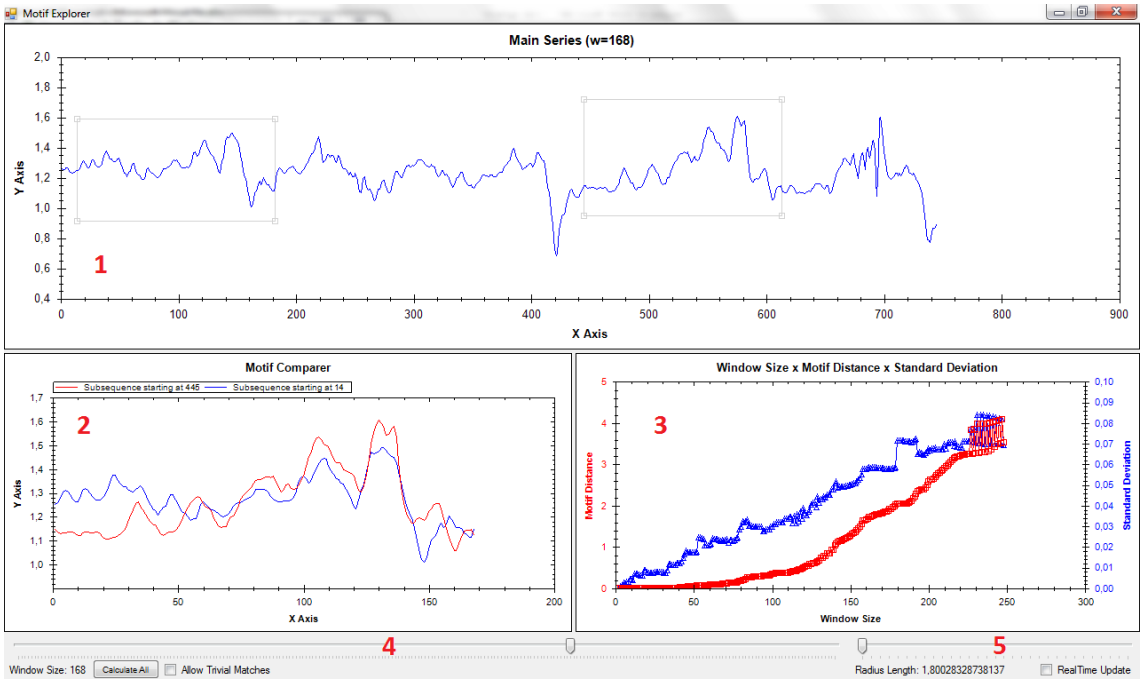

Figura 5.5: Interface da implementação baseada em Motifs.

5.3 SOM *versus* Motifs

Inicialmente utilizou-se a série das vazões do Rio São Francisco, com amostras diárias, ao longo de oito anos. Gerando-se o gráfico da série, percebe-se claramente que o gráfico é sazonal, podendo-se ter uma idéia prévia do tipo de padrão que se espera encontrar. A figura 5.6 exibe a série com os motifs identificados em vermelho.

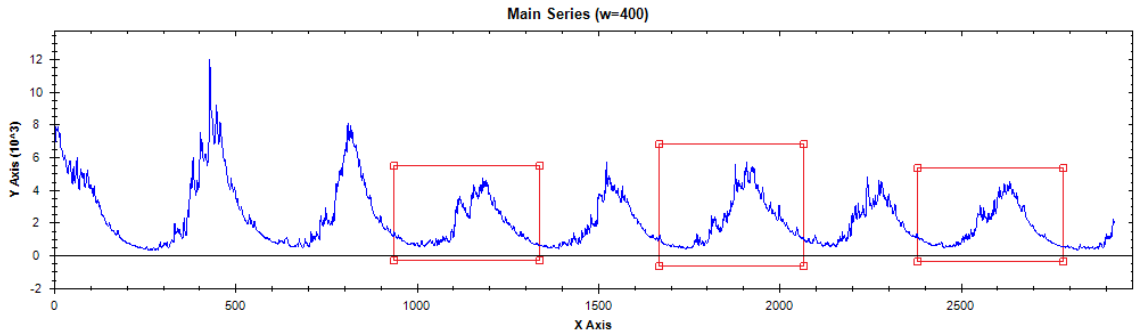

Figura 5.6: Um K-Motif capturado em uma série temporal.

As subsequências mais similares encontradas foram sobrepostas na figura 5.7, exibida a seguir, para facilitar a comparação. Tal grupo de subsequências corresponde ao 1-Motif da série, ou seja, o grupo com o maior número de elementos que distam entre si em no máximo uma distância *R*, desconsiderando-se os *trivial matches*.

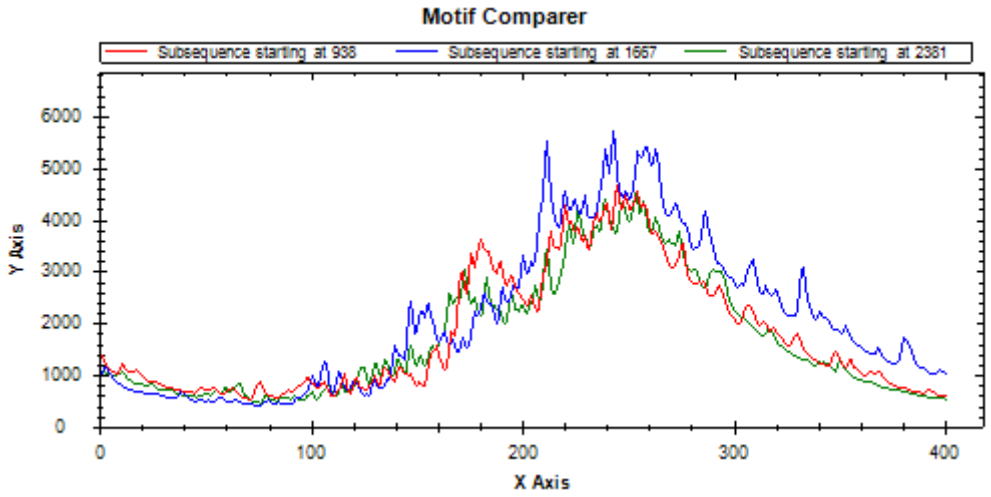

Figura 5.7: Subsequências do K-Motif sobrepostas para comparação.

Ao fornecer as subsequências encontradas como entrada para o k-*means*, com *k* igual ao número de K-Motifs encontrados, obtém-se *k* padrões, que correspondem à média das subsequências de cada grupo. A figura 5.8 exibe o padrão encontrado a partir das subsequências exibidas na figura 5.7. Ainda na figura 5.8, é interessante observar que o padrão encontrado não sofreu muita suavização, apresentando ainda um bom nível de detalhamento quando comparado à sequencia original. Isso se dá pelo fato da média ter sido calculada sobre subsequências muito semelhantes, que foram previamente selecionadas durante a fase de descoberta de motifs.

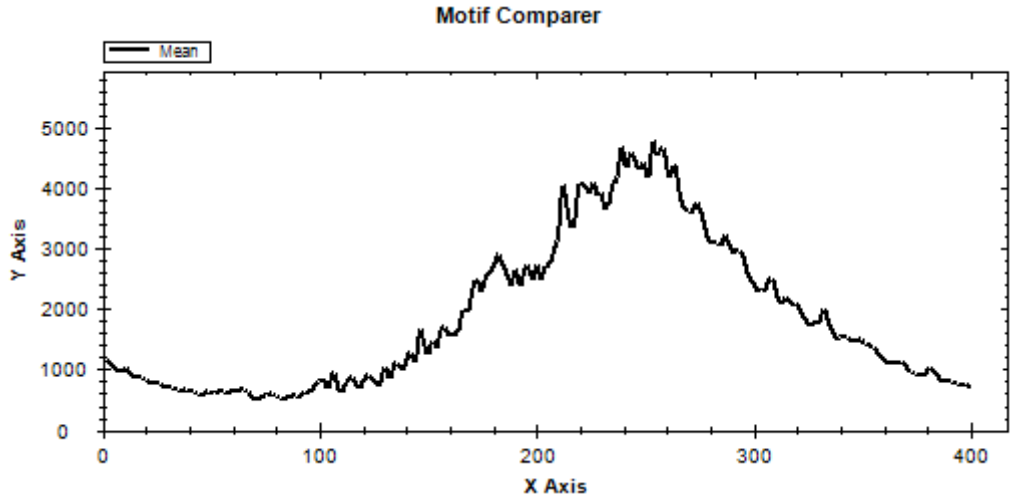

Figura 5.8: Média das subsequências pertencentes ao K-Motif.

Por outro lado, executando-se a implementação do SOM com a modificação que desconsidera *trivial matches* baseada em PIPs (CHUNG *et al*, 2005), sobre a mesma série de dados, obtém-se na camada de saída os padrões exibidos a seguir (figura 5.9).

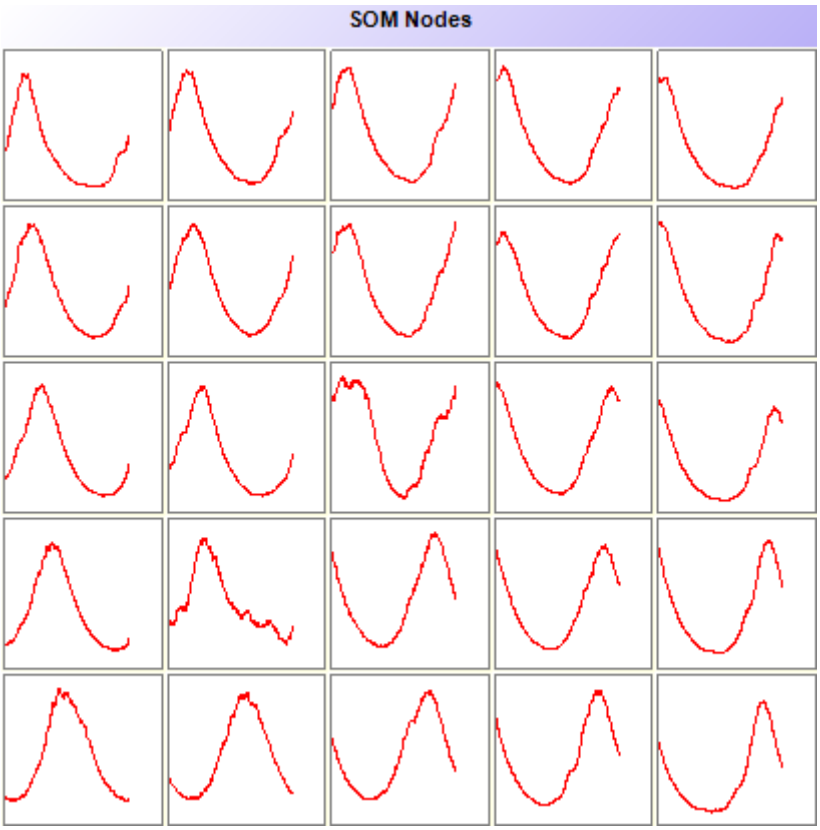

Figura 5.9: Padrões gerados na saída do SOM.

Na figura 5.9, é possível notar que a obrigatoriedade em se especificar *a priori* uma quantidade de padrões na camada de saída gera muita redundância, o que justifica a posterior etapa de remoção de redundância. Após realizada a etapa de agrupamento chegou-se aos padrões exibidos na figura 5.10.

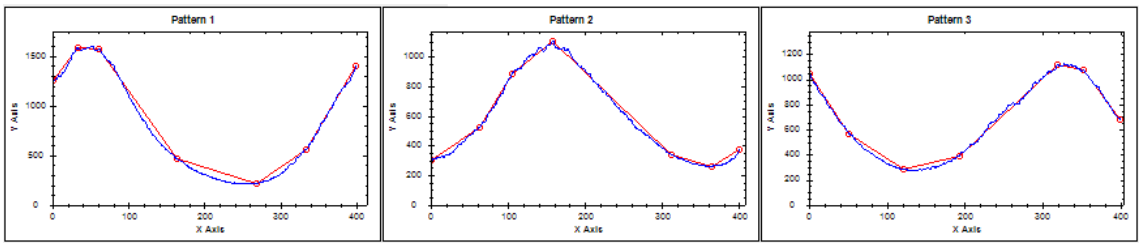

Figura 5.10: Padrões gerados pelo agrupamento da camada de saída do SOM.

Os padrões encontrados referem-se, na realidade, ao mesmo padrão, diferindo por estarem deslocados no tempo. Torna-se interessante, então, aumentar ainda mais a constante de agrupamento, uma vez que a intuição visual indica que a série possui um

único padrão. No entanto, diferentemente do que ocorreria no K-*means*, onde o agrupamento em um mesmo cluster se dá pela média e levaria a um padrão constante, Chung *et al* (2001) sugeriram um agrupamento onde a contribuição das componentes fosse ponderada pelo número de ocorrências de cada padrão na série original. Com isso, chega-se ao padrão exibido na figura 5.11.

É interessante notar que os valores do domínio foram bastante distorcidos. Enquanto que na série original os picos estavam acima de 4.000, nos candidatos a padrões finais obtidos pelo SOM os valores não passam de 1.000. Além disso, a suavização gerada na curva é tal que faz com que o resultado encontrado não remeta visualmente às subsequências da série original.

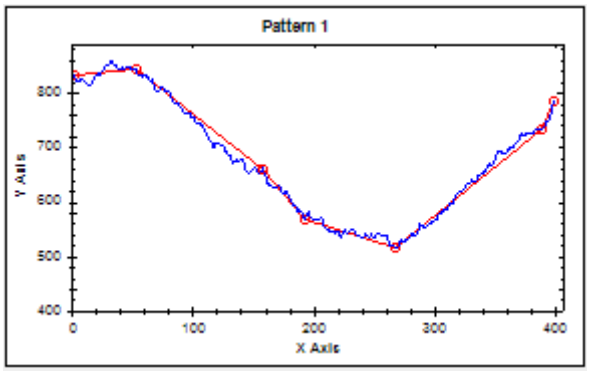

Figura 5.11: Padrão final encontrado com o SOM.

A figura 5.12, exibida a seguir, mostra a série original (A), e confronta os padrões encontrados pelas técnicas dos Motifs (B) e SOM (C). As subsequências destacadas correspondem ao 1-Motif da série original, que deram origem ao padrão mostrado na Figura 5.12B.

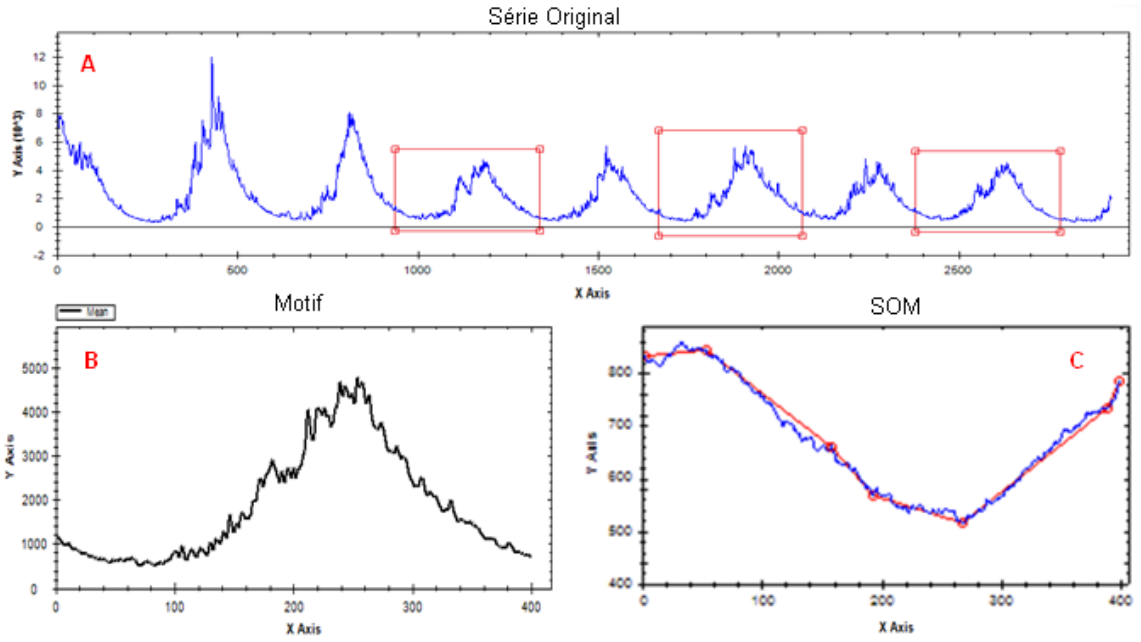

Figura 5.12: Série Original (A) e os padrões encontrados pelas abordagens Motif (B) e SOM (C).

Ao confrontar os resultados, é possível notar que a abordagem da busca por Motifs encontrou um padrão mais bem detalhado. Isso se deve ao fato da média ser calculada sobre um número menor de subsequências – aquelas que pertencem ao 1-Motif. A técnica do SOM, por outro lado, produziu um padrão mais suavizado, atenuando os picos que se repetem ao longo da série original e distorcendo significativamente os valores do domínio, representados no eixo y. Além disso, o esforço para se obter o padrão baseado em motifs se mostrou menor, tanto em processamento quanto em experimentação de parâmetros.

6 CONSIDERAÇÕES FINAIS

Apesar de a busca de padrões em séries temporais ser de grande importância para o estudo em diversas áreas, não se estabeleceu um consenso sobre qual é a melhor técnica a se aplicar. No trabalho de Keogh e Lin (2003), onde citou-se diversos estudos que utilizavam a abordagem de janela deslizante com *clustering*, esse parecia ser um método largamente utilizando entre os anos de 1998 e 2003, variando-se apenas a forma como era feita a discretização da série e o algoritmo de *clustering* escolhido. Após a afirmação que o método gerava padrões aleatórios, muitos dos autores que tiveram seus trabalhos citados admitiram realmente que a abordagem apresentava problema; outros, no entanto, buscaram soluções alternativas que tentassem contornar o problema através do pré-processamento das entradas, como em Chung *et al* (2005).

Pôde-se perceber através das implementações realizadas ao longo desse estudo que tanto o algoritmo SOM, quanto o K-*means* cumprem muito bem a tarefa de realizar *clustering* de elementos, ou *whole clustering* de séries temporais. O problema surgiu quando se passou para tais algoritmos, subsequências geradas pela janela deslizante, que fez com que os padrões obtidos pouco lembrassem a série original, por serem esses muito suavizados e de baixa resolução. Ao utilizar o SOM na etapa de *clustering* percebeu-se que, ao realizar o processo de remoção de redundância sugerido por Chung *et al* (2001), os padrões agrupados tendiam a produzir curvas cada vez mais suavizadas, semelhantes aos senos degenerados mencionados em Keogh *et al* (2003). Isso leva ao problema de definir o também o valor ideal para a constante de agrupamento, além de todos os outros parâmetros do SOM.

A abordagem baseada em Motifs, sugerida inicialmente por Keogh e Lin (2003) e mais tarde aperfeiçoada em Keogh *et al* (2009), gerou resultados melhores. Com um princípio mais simples e mais bem formalizada, a técnica gera padrões de melhor resolução e que são mais facilmente identificados na série original. Isso se deve ao fato do algoritmo buscar subsequências similares, em vez de realizar cálculos de ajustes baseados na média das subsequências, como ocorre, por exemplo, ao se utilizar janela deslizante com o algoritmo K-*means*. A abordagem sugerida pelos autores, além de produzir resultados qualitativamente mais significativos, se mostrou mais rápida que o SOM e com menos parâmetros a configurar.

Portanto, o SOM, apesar gerar padrões pouco significativos, consegue capturar as tendências gerais das séries, eliminando as variações instantâneas, caracterizadas como ruídos. A abordagem baseada em Motifs, por outro lado, considera todos os elementos da série ao buscar por subsequências semelhantes, incluindo o que seria o ruído aleatório fazendo com que os padrões encontrados sejam mais ricos em detalhes. Isso faz com que essa última abordagem seja mais apropriada à busca de padrões em subsequências de séries temporais, principalmente quando se está interessado em estudar variações em alta resolução, enquanto que a técnica de *clustering* fica mais

adequada ao *whole clustering* (*clustering* de sequencias inteiras) ou quando se está interessado em capturar tendências.

A busca de padrões baseada em motifs se mostrou, portanto, mais adequada ao tipo de resultado que se buscava, ao produzir padrões semelhantes ao das séries de entrada. Além disso, possibilitou o desenvolvimento de uma implementação eficaz no que diz respeito ao uso dos recursos de processamento, e com menos parâmetros a serem configurados, quando comparada ao SOM.

6.1 Melhorias e Trabalhos Futuros

O algoritmo de identificação de motifs sugerido em Keogh *et al* (2009), cumpre bem a tarefa de identificar o par de subsequências mais similares em uma série temporal, no entanto, para se encontrar o maior grupo de subsequências que distam entre si em no máximo uma distância R – referenciado no texto por 1-Motif – tal algoritmo não auxilia. Isso ocorre porque o par mais similar de subsequências pode não fazer parte do grupo mais significante, e qualquer tentativa de partir desse algoritmo levaria a uma solução gulosa, onde sempre buscar pelo padrão mais similar partindo do motif inicial, gera um máximo local. Buscar a solução ótima, nesse caso, trata-se um problema combinatório, e poderia ser resolvido através de programação inteira.

Na implementação do SOM, pode ser interessante visualizar os padrões resultantes ordenados por significância, ou seja, aqueles que recebem mais ocorrências das subsequências utilizadas no treinamento da rede. Além disso, realizar mais testes com o número de nós do SOM maior que a janela parece, apesar de custoso, uma opção que uma opção que leva a resultados melhores. Outro ponto interessante para o estudo, a fim de avaliar a forma como os padrões foram agrupados durante o processo de remoção de redundância na saída do SOM, torna-se interessante a geração de um dendograma ao longo do processo.

Para comprovar a significância dos resultados obtidos por Chung *et al* (2005), onde sugeriu-se utilizar o processo de identificação de PIPs para evitar os *trivial matches*, seria interessante refazer o teste de significância realizado por Keogh *et al* (2003). Por fim, em séries sazonais, poderia se calcular o coeficiente de autocorrelação da série, a fim de auxiliar na definição de um tamanho inicial w da janela deslizante.

REFERÊNCIAS

CHUNG, F. L.; Fu, T. C.; Ng, V.; Luk, R. Pattern Discovery from Stock Time Series Using Self-Organizing Maps. **The 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001) Workshop on Temporal Data Mining**, San Franscisco, Califórnia, p. 27-37, Ago. 2001.

CHUNG, F. L.; Fu, T. C.; Ng, V.; Luk, R. Flexible Time Series Pattern Matching Based on Perceptually Important Points. **International Joint Conference on Artificial Intelligence (IJCAI'01) Workshop on Learning from Temporal and Spatial Data**, Seattle, Washington, p. 4-10, Ago. 2001a.

CHUNG, F. L.; Fu, T. C.; Luk, R; Ng, C. M. Preventing Meaningless Stock Time Series Pattern Discovery by Changing Perceptually Important Point Detection. **The 2nd International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2005)**, Changsha, China, p. 27-29, Ago. 2005.

FU, T. C. A review on time series data mining. **Engineering Applications of Artificial Intelligence ELSEVIER 2010**, Hong Kong, Fev 2008.

ASTROM, K. J. On the choice of samping rates in parametric identification of time series. **Information Sciences 1 (3)** [S.l.:s.n], p. 273-278, 1969.

Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M. Locally adaptive dimensionality reduction for indexing large time series databases. **Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data**, Califórnia, p. 151-163, Mai. 2001.

KEOGH, E.; Lin, J. Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research. **Proceedings of the 3rd IEEE International Conference on Data Mining**, Melbourne, p. 154-177, Nov. 2003.

CHIU, B.; Keogh, E., Leonardi, S. Probabilistic Discovery of Time Series Motifs. **The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**, Washington, p. 493-498, Ago. 2003.

BRADLEY, P. S.; Fayyad, U. M. Refining Initial Points for K-Means Clustering. **In proceedings of the 15th Int'l Conference on Machine Learning**. Madison, p. 91-99, Jul. 1998.

IDÉ, T. Why Does Subsequence Time-Series Clustering Produce Sine Waves? **In Proc. European Conf. Principles and Practice of Knowledge Discovery in Databases, Lecture**. Yamato, p. 211-222, 2006.

MUEEN, A.; Keogh, E.; Zhu, Q.; Cash, S.; Westover, B. Exact Discovery of Time Series Motifs. **In SIAM International Conference on Data Mining (SDM09)**, [S.l.:s.n] Mai. 2009.

SHUMWAY, R.; STOFFER D. Time Series Analysis and Its Applications With R Examples. Springer, 2006.

BUCKLAND, M. Kohonen's Self Organizing Feature Maps. **AI – Junkie** [S.l.:s.n], Abril 2005. Disponível em: <http://www.ai-junkie.com/ann/som/som1.html>. Acesso em: Nov. 2010.

LANGAGER, C.; Murphy, C. Analyzing Chart Patterns: Head And Shoulders. **Investopedia** [S.l.:s.n], Jun 2006. Disponível em: <http://www.investopedia.com/university/charts/charts2.asp>. Acesso em: Nov. 2010.

GLOSSÁRIO

Série Temporal – conjunto ordenado de valores reais.

Subsequência – subconjunto contíguo de valores de uma série temporal.

Janela Deslizante – processo de obtenção de subsequências através do deslizamento de uma janela de tamanho fixo sobre a série temporal.

Data Mining – processo de explorar grandes quantidades de dados em busca de padrões consistentes.

Clustering – técnica de *Data Mining* utilizada para fazer agrupamento em dados, com base em algum critério de semelhança, onde tal critério faz parte da definição do problema.

Cluster – Agrupamento de elementos gerados por um algoritmo de *clustering*.

Motif – termo recorrente em áreas como música, literatura, artes visuais, têxtil e biológica, usado para se referir a padrões se repetem em uma sequencia.

Random Walk – processo de geração de uma série de dados através de um processo onde cada valor é gerado aleatoriamente, não sofrendo influência de seus sucessores ou antecessores.